НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

Кафедра обчислювальної техніки

«На правах рукопису»
УДК ___004.05____

До захисту допущено:
Завідувач кафедри
_____ Сергій Стіренко
«___»_____2021 р.

МАГІСТЕРСЬКА ДИСЕРТАЦІЯ

на здобуття ступеня магістра

за освітньо-науковою програмою «Комп'ютерні системи та мережі»

зі спеціальності 123 «Комп'ютерна інженерія»

на тему: «Метод цифрового голосування на основі технології блокчейн»

Виконав:
студент VI курсу, групи ІО-91мн
Бобовський Євген Олександрович _____

Керівник:
доцент кафедри ОТ, к.т.н., доцент,
Волокита Артем Миколайович _____

Консультант з нормоконтролю:
професор кафедри ОТ, д.т.н., професор,
Кулаков Юрій Олексійович _____

Рецензент
доцент кафедри АУТС, к.т.н., доцент
Катін Павло Юрійович _____

Засвідчую, що у цій магістерській дисертації немає запозичень з праць інших авторів без відповідних посилань.
Студент _____

Київ – 2021 року

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»

Факультет (інститут) *Інформатики та обчислювальної техніки*
(повна назва)

Кафедра *Обчислювальної техніки*
(повна назва)

Рівень вищої освіти – другий (магістерський) за освітньо-науковою програмою *«Комп'ютерні системи та мережі»*

Спеціальність *123. Комп'ютерна інженерія*
(код і назва)

ЗАТВЕРДЖУЮ
Завідувач кафедри
С.Г. Стіренко
(підпис) (ініціали, прізвище)
« » 20***21*** р.

ЗАВДАННЯ
на магістерську дисертацію студенту
Бобовському Євгену Олександровичу
(прізвище, ім'я, по батькові)

1. Тема дисертації *Метод цифрового голосування на основі технології блокчейн*
Науковий керівник дисертації *к.т.н., доц. Волокита А.М.*
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
затверджені наказом по університету від «*12*» *березня 2021* р. № *809-с*

2. Строк подання студентом дисертації *26 квітня 2021 року*

3. Об'єкт дослідження *процес цифрового голосування*

4. Предмет дослідження *методи та засоби підвищення ефективності цифрового голосування*

5. Перелік завдань, які потрібно розробити: *дослідити методи голосування, що існують; визначити базові вимоги до демократичного голосування; розробити метод, що відповідає визначеним вимогам; створити програмне забезпечення дошки оголошень на основі блокчейн; проаналізувати отримані результати*

6. Консультанти розділів дисертації:

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Нормоконтроль	Кулаков Ю.А., професор		

7. Дата видачі завдання _____

Календарний план

№ з/п	Назва етапів виконання магістерської дисертації	Строк виконання етапів дисертації	Примітка
1.	*Затвердження теми роботи*	*08.02 – 14.02.2021*	
2.	*Вивчення та аналіз завдання*	*15.02 – 21.02.2021*	
3.	*Аналіз описаних раніше методів цифрового голосування*	*22.02 – 07.03.2021*	
4.	*Визначення вимог до методу*	*08.03 – 22.03.2021*	
5.	*Проектування методу*	*23.03 – 04.04.2021*	
6.	*Програмна реалізація методу*	*05.04 – 18.04.2021*	
7.	*Оформлення пояснювальної записки*	*19.04 – 25.04.2021*	
8.	*Передзахист*	*26.04.2021*	
9.	*Захист*	*18.05.2021*	

Студент _____ *Є.О. Бобовський*
(підпис) (ініціали, прізвище)

Науковий керівник дисертації _____ *А.М. Волокита*
(підпис) (ініціали, прізвище)

АНОТАЦІЯ

У даній роботі описано метод цифрового голосування з частковою перевіркою на основі технології блокчейн. Метод призначений для голосування на виборчих дільницях. Метод базується на адитивному гомоморфному пороговому шифруванні та використовує мережі перемішування. У якості дошки оголошень для максимальної прозорості проведення голосування виступає блокчейн, до якого записуються усі операції, що виконують сервери. Метод відповідає основним вимогам до демократичного голосування. Часткова перевірка забезпечує аудит на кожному етапі голосування та дає впевненість, що будь-яке нелегальне втручання у хід голосування буде виявлене з високою ймовірністю.

ABSTRACT

This paper describes blockchain-based digital voting method with partial checking. The method is designed for voting at polling stations. This method is based on homomorphic threshold encryption and also uses mix nets. The method has a bulletin board based on blockchain, which contains all operations performed by servers, to provide as much transparency as possible. The described method meets all of the requirements for democratic voting. The partial checking provides audit on every stage of the election and guarantees with high probability that any error occurred during election will be spotted.

РЕФЕРАТ

на магістерську дисертацію

виконану на тему: Метод цифрового голосування на основі технології блокчейн

студентом: Бобовським Євгеном Олександровичем

Дисертацію виконано на 100 аркушах. Вона містить перелік посилань на використані джерела з 79 найменувань. У роботі представлено 16 рисунків.

Актуальність. Електронне голосування є гарячою темою для обговорення протягом кількох останніх десятиліть. У наш час досі тривають дискусії з приводу того, як краще організувати електронне голосування та чи варто його впроваджувати взагалі.

Попри численні дослідження та напрацювання, досі немає ідеального методу, який відповідав би всім необхідним вимогам для того, аби замінити паперове голосування.

Мета і завдання дослідження. Метою магістерської роботи є розробка методу цифрового голосування на основі технології блокчейн, який відповідає всім вимогам до демократичного голосування.

Для досягнення мети дослідження поставлено й вирішено завдання:

- дослідження методів голосування, що існують;
- визначення базових вимог до демократичного голосування;
- розробка методу, що відповідає визначеним вимогам;
- аналіз отриманого результату.

Об'єктом дослідження є процес цифрового голосування.

Предметом дослідження є методи та засоби підвищення ефективності цифрового голосування.

Методи досліджень. Для досягнення поставлених у магістерській роботі задач використано технологію блокчейн, за яку відповідає фреймворк Graphene, технологію цифрового підпису, криптосистему Пайє, гомоморфне шифрування, мережі перемішування.

Наукова новизна одержаних результатів. У даній магістерській роботі запропоновано метод цифрового голосування, який, за рахунок використання гомоморфного порогового шифрування криптосистеми Пайє, мереж перемішування та блокчейну в якості дошки оглошень, дозволяє забезпечити голосування без використання паперових бюлетенів, яке при цьому відповідає всім базовим вимогам до демократичного голосування.

Апробація результатів дисертації. Результати досліджень, що включені до дисертації, доповідались на І Міжнародній науковій спеціалізованій конференції «Сучасні напрямки розвитку автоматизації, транспортних систем, технічних та комп'ютерних наук»

Публікації.

1. Bobovskyi Y. Digital Voting / Yevhen Bobovskyi // Наука та техніка XXI століття / – Kyiv, 2020.
2. Бобовський Є. Метод цифрового голосування з частковою перевіркою / Євген Бобовський // Сучасні напрямки розвитку автоматизації, транспортних систем, технічних та комп'ютерних наук / – Полтава, 2021, DOI: https://doi.org/10.36074/mcnd-30.04.2021.engineering.03.

Ключові слова. Цифрове голосування, електронне голосування, гомоморфне шифрування, блокчейн, цифровий підпис, мережа перемішування.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

Бекенд – програмно-апаратна частина сервіса, яка відповідає за внутрішній функціонал

Біткоін – криптовалюта, у реалізації якої був вперше представлений блокчейн

Блок – одна або більше транзакцій, які об’єднані заголовком та захищені доказом виконання роботи

Блокчейн (англ. blockchain) – ланцюг блоків транзакцій; розподілена база даних

ВД – виборча дільниця

Вихід транзакції (англ. transaction output) – кількість сатоші, що використовуються у транзакції в якості вихідного параметру; кількість сатоші що отримують

ВК – виборча комісія

ВКУ, Кодекс – Виборчий кодекс України

Вхід транзакції (англ. transaction input) – кількість сатоші, що використовуються у якості вхідного параметру; кількість сатоші, що передають

ДВК – дільнична виборча комісія

Дерево Меркла (англ. Merkle tree) – дерево, що утворене шляхом попарного хешування даних (листів; англ. leaves), а потім попарного хешування хешів, допоки не залишиться один хеш

Доказ виконання роботи (англ. proof of work; pow) – принцип захисту, що полягає у необхідності виконання однією стороною досить складної операції обчислення, у той час як інша сторона може з легкістю перевірити правильність результату

ДРВ – державний реєстр виборців

Майнер (англ. miner) – пристрій, що додає нові блоки до блокчейну або власник даного пристрою

Невитрачений вихід транзакції (англ. Unspent Transaction Output, UTXO) – вихід транзакції, який ще не було використано у якості входу

Сатоші (англ. satoshi) - найменша одиниця криптовалюти біткоіну; $1/10^8$ біткоіну

ТВК – територіальна виборча комісія

ЦВК – Центральна виборча комісія

Advanced Encryption Standard (AES) – блочний симетричний алгоритм шифрування, також відомий як Rijndael

Delegated Proof of Stake (DPoS) – алгоритм, що використовується замість PoW у Graphene

Proof of Work (PoW) – алгоритм захисту системи від зловживання послугами

Rivest-Shamir-Adleman (RSA) – асиметричний алгоритм шифрування

ВСТУП

Демократичне голосування є серйозною, а у деяких випадках навіть вирішальною, подією у будь-якій країні. Зі стародавніх часів люди користувались голосуванням для прийняття рішень. Для цього вони піднімали руки, кидали різнокольорові кульки в чашу чи навіть просто кричали. Найбільш популярним методом голосування зараз є стара паперова система, якою користуються вже доволі давно.

Цифрове голосування – використання електронних пристроїв (від спеціальних машин до персонального комп'ютера) для того, аби віддати свій голос. Його також іноді називають електронним голосуванням або е-голосуванням (e-voting) коли використовують спеціальну машину та і-голосуванням (i-voting) під час використання веб-браузера.

Очевидно, що головним занепокоєнням під час розгляду можливої реалізації подібної системи є її безпека. Не можна покладатись на систему для вирішення настільки важливих питань, якщо немає впевненості у її захищеності та здатності вистояти проти можливих атак.

Найбільшим викликом для методів цифрового голосування стало забезпечення двох практично взаємновиключних умов:

- виборець повинен бути впевненим у тому, що його голос було зараховано і не було змінено;
- виборець не повинен мати змоги надати докази того, як він проголосував.

Ця, та ще низка проблем, спонукали багатьох фахівців відноситись скептично до цифрового голосування [22, 30, 9, 58], стверджуючи, що перехід від паперового до цифрового голосування не є доцільним та створює нові ризики.

Утім, інші дослідники не втрачали оптимізму та пропонували різноматні підходи до вирішення завдання [35, 57, 36, 21]. Не всі з них задовольняли всі необхідні вимоги, та були далеко не ідеальними. Так, наприклад, два з

найбільш вдалих методів: Scratch & Vote [7] та PrÊt À Voter [50] призначені лише для голосування на виборчих дільницях та все ще базуються на паперових бюлетенях. Окрім того, вони трохи ускладнюють алгоритм голосування для виборця та надають йому частину бюлетеню як підтвердження голосу, завдяки якому він хоч не може довести нікому як проголосував, однак це відкриває можливість для кількох атак на легітимність виборів.

Практично в усіх запропонованих на сьогодні методах голосування присутнє поняття публічної дошки оголошень, яка необхідна для прозорості проведення голосування. При цьому інформацію на цю дошку можна тільки записувати та зчитувати з неї. Видаляти чи редагувати інформацію не можна.

Окрім того, у більшості методів беруть участь декілька сторін, які відповідають за різні функції під час голосування (реєстрація виборців, прийом, шифрування, підрахунок голосів тощо).

Саме тому у даній роботі використано технологію блокчейн [13], з допомогою якої організовано розподілену систему голосування, роботу якої забезпечує множина сторін, що не зацікавлені у співпраці з одне одним, адже є представниками різних таборів на голосуванні. Більш того, блокчейн добре підходить для реалізації дошки оголошень, адже володіє усіма необхідними характеристиками.

РОЗДІЛ 1
АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Паперове голосування

Розглянемо традиційне паперове голосування на прикладі місцевих виборів в Україні.

Голосування на виборах в Україні є таємним. Контроль за змістом волевиявлення виборців, встановлення або розголошення змісту волевиявлення конкретного виборця будь-яким чином забороняється [2].

Кожен виборець голосує на виборах особисто [2]. Голосування за інших осіб чи передання виборцем права голосу будь-якій іншій особі забороняється.

Допомога виборцю, який внаслідок інвалідності та/або стану здоров'я не може самостійно заповнити виборчий бюлетень чи опустити його у виборчу скриньку, не вважається порушенням, якщо така допомога надана відповідно до його волевиявлення [2].

Виборчою дільницею називається територіальна одиниця, яка утворена для організації і проведення голосування. Виборчі дільниці поділяють на звичайні та спеціальні, які знаходяться у заходах охорони здоров’я. У залежності від кількості виборців, що закріплені за ВД, поділяються на малі (до 500 осіб), середні (500-1500 осіб) та великі (від 1500 осіб). Розміри ВД визначають відповідне забезпечення ДВК необхідними ресурсами [4].

Напередодні голосування ДВК отримує попередні списки виборців, які подає у своєму приміщенні для загального ознайомлення. У цей список можуть бути внесені зміни за зверненням виборця або за рішенням суду [4]. Попередні списки виборців на спеціальних дільницях складають у відповідності до звернення керівника закладу, у якому розміщена ця дільниця.

Особам, які з об’єктивних причин не можуть прийти на виборчу дільницю, надається змога проголосувати «за місцем перебування».

Бюлетені прибувабть на виборчу дільницю не раніше ніж за 3 дні до початку голосування. Після прибуття всі бюлетені розпаковуються, перераховуються, поміщаються до сейфу та опечатуються.

Обладнання приміщення для голосування є відповідальністю ДВК [2]. У цьому приміщенні повинні бути [4]:

- місця видачі виборчих бюлетенів,
- виборчі скриньки
- кабіни для голосування (принаймні одна з яких повинна бути обладнана для голосування виборців з порушенням здоров'я)

Голосування проводиться з 8:00 до 20:00. Не раніше як за 45 хв. до початку голосування розпочинається підготовче засідання ДВК, під час якого відбувається розподіл обов'язків членів комісії та останні підготування до проведення голосування.

Виборець повинен самостійно заповнити бюлетень та опустити його в скриньку. У разі порушення здоров'я унаслідок якого зробити це самостійно неможливо, виборець може попросити іншого виборця допомогти йому заповнити бюлетень та опустити його в скриньку, попередньо повідомивши члена ДВК [2].

У разі зіпсування виборцем бюлетеня, він може звернутись до ДВК з проханням замінити зіпсований бюлетень на новий. Зіпсований бюлетень одразу погашається, про що складається акт [4].

У разі пошкодження скриньки для голосування, вона опечатується таким чином, щоб унеможливити подальше опускання чи виймання з неї бюлетенів, та зберігається у місці, що знаходиться під наглядом ДВК [4].

За 5 хвилин до 20-ої години голова ДВК оголошує про закінчення голосування та зачинення ВД о 20:00. При цьому один з членів комісії рівно о 20 годині заводить усіх виборців, що станом на 20 годину прийшли до ВД, та зачиняє за ними двері. Голосування закінчується коли з приміщення виходить останній виборець. У приміщенні ВД залишаються тільки члени ВК та особи, які мають право бути присутніми на засіданні ВК [4].

Голосування виборців за місцем перебування проводять не менш як 3 члени ДВК. Їх робота організовується таким чином, щоб вони повернулись до ВД не пізніше 19:00 [2]. Перед відбуттям членів ДВК для організації голосування за місцем перебування, голова ДВК видає їм витяг зі списку виборців, примірники інформаційних плакатів та буклетів та опечатану скриньку. У скриньку вкладається контрольний лист, у якому зазначається інформація про виборчу скриньку, членів ДВК, час їх виходу та кількість отриманих бюлетенів. Контрольний лист підписується присутніми членами ДВК [4].

Якщо виборець, що мав голосувати за місцем перебування, прибув на виборчу дільницю, то йому не можуть видати бюлетень до повернення членів комісії, що організовують голосування за місцем перебування, аби впевнитись що виборець не збирається проголосувати двічі [2].

До відкриття виборчих скриньок проводиться фіксація кількості виборців, що взяли участь у голосуванні, підраховуються невикористані бюлетені [2].

Контрольні талони запаковуються в окремі пакети з написом “контрольні талони” [4].

Після цього ДВК запаковує в окремий пакет список виборців, витяг зі списку виборців, заяви, на підставі яких складався витяг, рішення суду про внесення змін до списку та повідомлення органу ведення ДРВ [2].

Бюлетені зі скриньок не підлягають врахуванню у випадках [2]:

- Якщо зазначені на бюлетені позначення виборчого округу або дільниці не відповідають тому, на якому проводиться підрахунок.
- У разі відсутності у скриньці контрольного листа.
- Якщо при відкритті переносної скриньки у ній виявиться більше виборчих бюлетенів з будь-яких із виборів, що проводяться одночасно, ніж кількість виборців, що включені до витягу зі списку.
- У разі сумнівів щодо достовірності контрольного листа.

Виборчі бюлетені, що не підлягають врахуванню, запаковуються з позначкою “виборчі бюлетені, що не підлягають врахуванню” [4].

Окремо підраховуються недійсні виборчі бюлетені. Недійсним бюлетень може бути визнаним якщо [4]:

- відсутня печатка ДВК;
- проставлено штамп “вибув” без відповідного рішення комісії вищого рівня, або не проставлено штамп, коли таке рішення було прийнято;
- проставлено декілька або жодної позначки;
- не відокремлено контрольний талон;
- неможливо встановити зміст волевиявлення.

Усі недійсні виборчі бюлетені запаковуються з позначкою “недійсні виборчі бюлетені” [4].

1.2 Scratch & Vote

Scratch & Vote – криптографічний метод голосування, який забезпечує публічний аудит. Він використовує паперові бюлетені, голос у яких відмічається з допомогою ручки. При цьому бюлетень містить усю необхідну інформацію для аудиту (рис 1.1) [7].

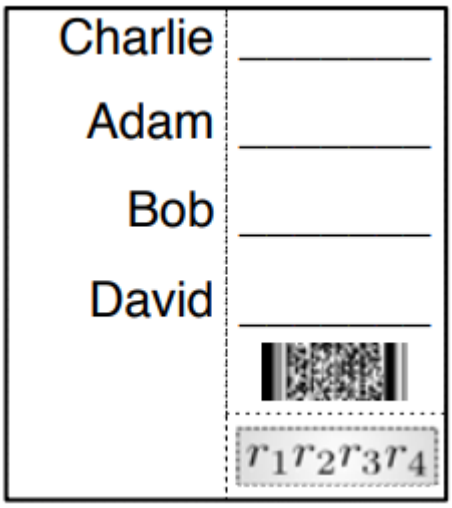

Рис. 1.1 Бюлетень Scratch & Vote

Бюлетень містить список кандидатів, що розміщений у випадковому порядку, ідентифікатор та дані, з допомогою яких було перемішано список. Ці дані приховані під спеціальним шаром, який можна здерти з допомогою монетки чи іншого підручного засобу.

Окрім цього, підрахунок голосів відбувається з використанням гомоморфного шифрування, а сторони, що беруть участь в організації голосування, повинні розшифрувати лише один шифротекст на одне голосування.

Таким чином, голосування проходить за таким алгоритмом:

1) виборець приходить на виборчу дільницю, реєструється та отримує два запечатаних бюлетені;
2) виборець прямує до кабінки для голосування, там розпаковує бюлетені та робить позначку в одному з них;
3) виборець відриває від бюлетеню ліву частину зі списком кандидатів та знищує її;
4) виборець демонструє свій бюлетень без списку кандидатів відповідальній особі з виборчої комісії, яка перевіряє цілісність шару, що приховує дані для відновлення послідовності кандидатів, відриває його від бюлетеню та знищує;
5) виборець поміщає свій бюлетень до сканеру;
6) виборець прибирає захисний шар зі свого другого бюлетеню, після чого може самостійно або з чиєюсь допомогою впевнитись у тому, що його було сформовано правильно;
7) виборець може підтвердити, що його голос було зараховано з допомогою свого браузера та правої частини бюлетеня.

Після завершення голосування усі голоси, завдяки адитивній гомоморфності алгоритму шифрування, який використовується, сумуються у зашифрованому вигляді. Результат голосування розшифровується кількома сторонами, що організовували голосування, з допомогою схеми порогового шифрування.

1.3 PrÊt À Voter

PrÊt À Voter – метод цифрового голосування, який у своїй основі має принципи, схожі з описаним вище. Він також базується на паперових бюлетенях, які містять необхідну інформацію для аудиту [50].

Метод складається із чотирьох складових: генерації бюлетенів, збору голосів, обробки голосів та аудиту.

Бюлетень, як і в попередньому методі, складається з лівої частини, де розміщені кандидати у порядку, згенерованому випадковим чином, та правої, де виборець повинен поставити позначку. Окрім цього, на лівій стороні бюлетеня міститься інформація, необхідна для відтворення списку кандидатів, яка зашифрована з допомогою порогового шифрування. З історичних причин ця інформація називається «цибулею» (onion), адже у перших версіях алгоритму вона розшифровувалась поетапно, шар за шаром. Бюлетень зображений на рис. 1.2.

Donald	
Barack	
Alice	
Crystal	
Edward	
	a6Gq21p

Рис. 1.2 Бюлетень PrÊt À Voter

Саме порядок кандидатів гарантує таємницю голосу. Ліва сторона відділяється від бюлетеню та знищується перед тим, як його буде відскановано, а праву частину виборець залишає собі. Тому лише виборець знає порядок кандидатів у бюлетені.

Збір голосів заключається у скануванні правої частини бюлетеня. Єдина необхідна криптографічна операція тут – це цифровий підпис виборця. Голоси

зберігаються у базі даних та публікуються на дошці оголошень, аби кожен виборець міг впевнитись у тому, що його голос враховано.

На етапі обробки голосів перед системою стоїть задача розшифрування та підрахунку голосів. Але це необхідно реалізувати таким чином, аби ніхто не зміг встановити відповідність між зашифрованим та розшифрованим голосами. Тож необхідно виконати три дії: перемішування, розшифрування, підрахунок. Для цього у даному методі використовується мережа перемішування, після проходження якої встановити відповідність між зашифрованим та розшифрованим голосом неможливо.

Аудит проводиться на кожному етапі голосування. Кожен виборець може впевнитись у тому, що його голос було зараховано. Кожна криптографічна операція супроводжується інформацією, якої достатньо для того, аби кожен бажаючий міг переконатись у правильності виконання операції.

1.4 Інтернет-голосування в Естонії

Естонія має систему цифрового голосування з 2005 року, і в 2007 стала першою країною, що дозволила онлайн голосування [38]. А вже на виборах 2019 року більше половини від усіх голосів було віддано через національну систему і-голосування [74].

Основою цієї системи є національне посвідчення особи, яке видається всім громадянам Естонії [25]. Це посвідчення має вигляд картки, з якою можна працювати з допомогою спеціального зчитувача та програмного забезпечення. Кожна така картка містить дві пари RSA ключів: одну для автентифікації і одну для цифрового підпису. Сертифікати, що підтверджують зв’язок між парами ключів та особою, що ними користується, зберігаються на самих картках та в публічній базі даних. Усі операції з використанням ключів виконуються на самій картці, тож жодне програмне забезпечення не має до них доступу. Також для використання ключу необхідний PIN [60].

Перед початком голосування Державна виборча служба Естонії готує систему та розповсюджує клієнт, що необхідний для голосування (рис. 1.3) [39]. Електронне голосування відкрите з 10-го до 4-го дня перед голосуванням. Для і-голосування необхідний комп'ютер, під'єднаний до мережі Інтернет, та національне посвідчення особи, мобільний ідентифікатор (для якого необхідна спеціальна SIM-карта [25]) або цифровий підпис. Голосування з допомогою інших розумних пристроїв (смартфони, годинники тощо) неможливе [73]. Необхідно завантажити на свій комп'ютер додаток для голосування, який автоматично перевірить право голосу виборця та відобразить список кандидатів.

Рис. 1.3 Клієнт для голосування в Естонії [39]

Для голосування з посвідченням особи необхідні [51]:

- національне посвідчення особи;
- PIN, який надається разом з посвідченням;
- діючі сертифікати;
- зчитувач карток, який можна придбати у будь-якому комп'ютерному магазині;

- програмне забезпечення для роботи з посвідченням та зчитувачем.

Для голосування з допомогою мобільного ідентифікатора [51]:

- SIM-карта, що підтримує ідентифікацію;
- PIN, який надається разом з SIM-картою;
- активувати мобільну ідентифікацію;

Програма-клієнт шифрує голос виборця, який підтверджує свій голос з допомогою обраного ним ідентифікатора, після чого голос відправляється на сервер збору голосів. Окремий незалежний сервер зберігає часові мітки для кожного голосу. Виборець може перевірити, чи його голос було отримано, за допомогою додатку через телефон (рис. 1.4) [39].

Після відправки свого голосу виборець може змінювати його необмежену кількість разів просто повторюючи процедуру голосування. Ураховуватись буде тільки останній поданий голос. Цифрове голосування закінчується до початку паперового і у виборця є можливість проголосувати з допомогою класичного паперового бюлетеня. Це автоматично анулює його і-голос [39].

Рис. 1.4 Перевірка голосу через мобільний додаток в Естонії [39]

Перед підрахунком над голосами здійснюють такі дії [39]:

1) анулюють усі голоси, що повинні бути анульованими;
2) відділяють персональні дані виборців від самих голосів;
3) розшифровують вміст голосів;
4) підраховують голоси.

Код системи і-голосування Естонії знаходиться у відкритому доступі в GitHub репозиторії [41].

Інфраструктура для голосування, представлена на рис. 1.5, складається з чотирьох машин [60]:

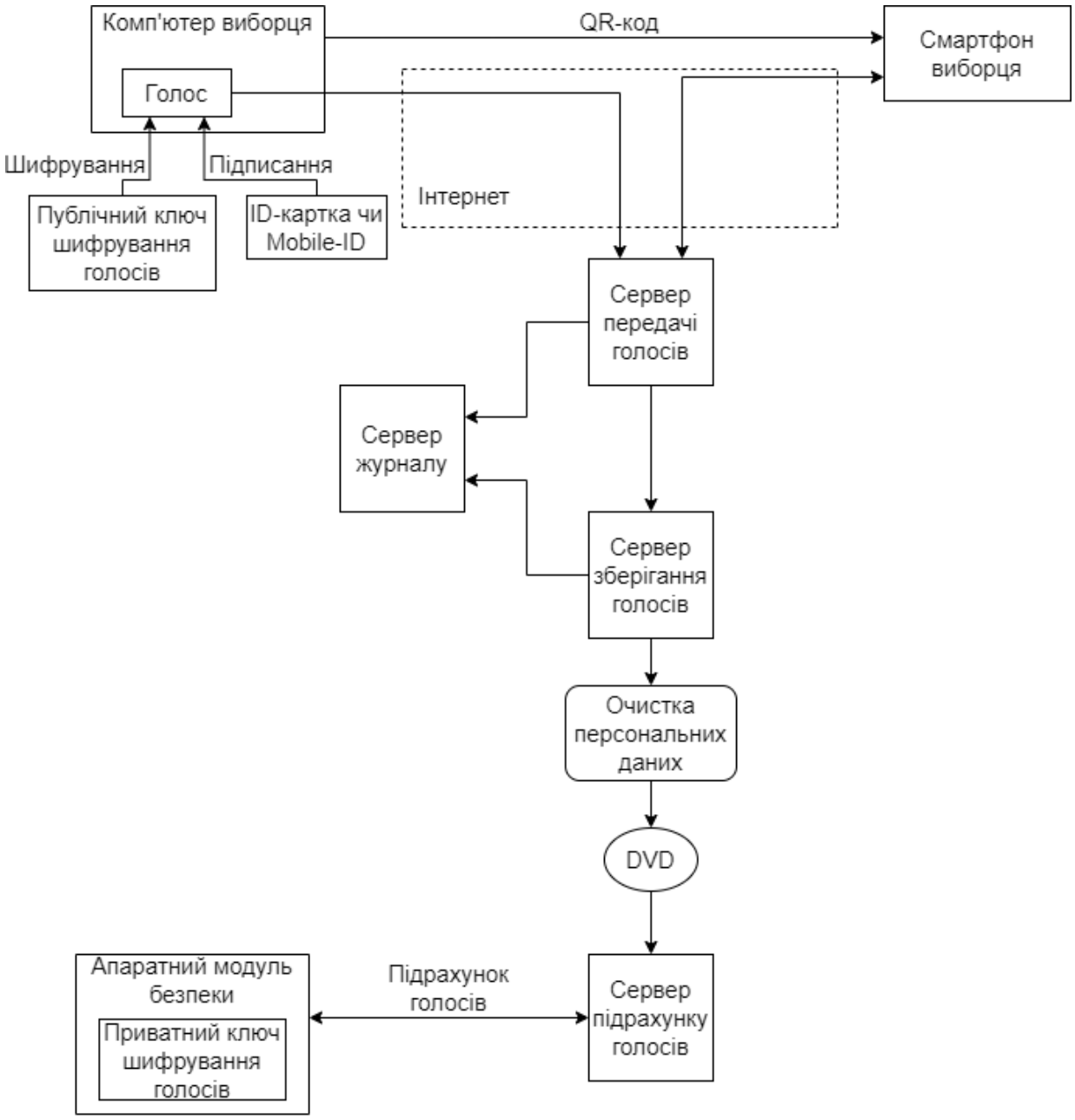

Рис. 1.5 Інфраструктура і-голосування Естонії

● Сервер відправки голосів (англ. Vote Forwarding Server, VFS; ест. HES) - єдиний сервер, доступний в мережі Інтернет. Він приймає HTTPS [52] запити від клієнтів голосування та переправляє їх на сервер зберігання голосів.

● Сервер зберігання голосів (англ. Vote Storage Server, VSS; ест. HTS) зберігає зашифровані та підписані голоси. Після отримання голосу від VFS перевіряє його формат та дійсність.

● Сервер журналу (англ. Log server) веде моніторинг, журнал подій та статистику, які отримує від VFS та VSS. Код цього серверу не було опубліковано.

● Сервер підрахунку голосів (англ. Vote Counting Server, VCS; ест. HLR) не під'єднаний до інфраструктури. Голоси потрапляють на нього шляхом копіювання їх на DVD. Сервер має апаратний модуль безпеки, який містить ключ для розшифровки голосів і використовується для підрахунку.

Під час місцевих виборів 2013 року дослідники спостерігали та вивчали процес голосування, висвітлюючи низку потенційних ризиків для безпеки системи [60].

Доповідь дослідників була розкритикована Естонським управлінням інформаційних систем [24].

1.5 Сучасні підходи з використанням блокчейну

Технологія блокчейн, з того самого моменту як побачила світ, прикувала до себе багато уваги. Це породило значну кількість проектів та стартапів з її використанням. Очевидно, що використання блокчейну для голосування не є новою ідеєю. Декілька країн уже намагались експериментувати з виборами використовуючи блокчейн.

Так, у 2018 році у штаті Західна Вірджинія, США, для надання можливості голосування військовим, які проходили службу за межами країни, було використано мобільний додаток "Voatz", що розроблений на основі технології блокчейн [79, 75].

У вересні 2019 року в Москві, Росія, під час проведення виборів у міську Думу, громадяни, на ряду з традиційним голосуванням, могли проголосувати онлайн якщо вони належали до трьох експериментальних виборчих дільниць [6].

Також голосування з використанням блокчейну проводили Японія [12] та Швейцарія [68].

Розглянемо те, що відомо про системи, які використовувались під час голосувань у Західній Вірджинії та Москві. Сама технологія блокчейн описана у наступному розділі.

1.5.1 Voatz

Мобільна платформа для голосувань “Voatz” увійшла в історію як перший додаток для голосування, що був використаний на федеральних виборах США [66].

Система використовує саме мобільні пристрої, адже вони мають більше механізмів безпеки, які можна використати у ході голосування. Голосування проходить за таким сценарієм [77]:

1) Попередня реєстрація: виборець реєструється як відсутній та завантажує додаток на телефон. Після цього підтверджує свою особу з допомогою будь-якого документа та прив’язує її до механізму захисту пристрою (відбиток пальця, PIN тощо). Копії документів, використаних для підтвердження особи, на серверах не зберігаються.

2) Голосування: з допомогою смартфону виборець віддає свій голос, підтверджуючи його обраним раніше способом.

3) Підтвердження голосу та аудит: виборець отримує квитанцію, яка підписана цифровим ідентифікатором. З допомогою цієї квитанції можна впевнитись, що голос виборця був отриманий та правильно записаний.

Голоси зберігаються як анонімні “транзакції голосування” у блокчейні.

Після здійснення виборцем волевиявлення, виборчий орган друкує бюлетень, який відповідає за цей голос. Цей бюлетень також підписаний з допомогою анонімного цифрового ідентифікатора, як і квитанція, що дозволяє перевірити легітимність голосу [77].

“Voatz” має систему захисту мобільних пристроїв від Zimperuim [78]. Ця система вбудована у мобільний додаток і при кожному його запуску проводить три перевірки [72]:

- перевірку вразливостей пристрою: сканує телефон на наявність вразливостей конфігурації (таких як “root-права” на Android [69] чи “jailbreak” на iOS [42]);
- перевірку додатків: система захисту сканує пристрій на наявність додатків, що не були підписані Apple чи Google;
- перевірку мережі: виконує моніторинг трафіку та помічає підозрілі з’єднання (наприклад, атаки “людина посередині”).

Усі з’єднання між пристроєм виборця та бекендом системи шифруються з використанням Advanced Encryption Standard (AES) [8], що працює в режимі Galois\Counter Mode (GCM) [23] з довжиною ключа 256 біт. Voatz використовує протокол Transport Layer Security версії 1.2 (TLSv1.2) [54] для забезпечення як з’єднання смартфон-сервер, так і сервер-сервер [72].

Пристрій кожного виборця створює пару приватного та публічного ключів. Сервер створює унікальну пару ключів для кожного виборця. Після чого сервер та пристрій обмінюються ключами використовуючи протокол Діффі-Геллмана на еліптичних кривих [26]. Увесь трафік відправляється через HTTPS [72].

На прикладному рівні “Voatz” поєднує SHA256 [53] з AES для шифрування смартфон-сервер та сервер-сервер повідомлень [72].

Для кожної сесії з’єднання використовуються унікальні пари ключів [72]. Це пов’язано з тим, що повідомлення у зашифрованому вигляді можуть бути збережені проміжними або ж кінцевими вузлами. У такому випадку усі

збережені повідомлення можна буде розшифрувати, якщо ключ стане відомим третій стороні.

Ключ для наступної сесії генерується і відправляється на пристрій під час поточної. Тобто, коли смартфон встановлює з'єднання з сервером, на перший раз сервер генерує випадкове значення ключа і зберігає його. Для всіх наступних з'єднань пристрій повинен використовувати попередньо згенерований сервером ключ, отриманий під час попередньої сесії [72]. Це дає змогу впевнитись що один обліковий запис не використовується на кількох пристроях одночасно.

Для зберігання голосів "Voatz" використовує Hyperledger Fabric, версію блокчейну з відкритим доступом [37]. Цей блокчейн спеціально розроблений для реалізації приватних блокчейнів, тобто таких, для доступу до яких необхідний дозвіл [72].

На цьому інформація, яку надає розробник " Voatz" закінчується. Тому, через те, що принципи роботи системи розкриті не повністю, до неї виникає низка запитань [76]. Зокрема, незрозуміло яким чином забезпечується таємність голосування.

Це, та ще низка сумнівів та занепокоєнь з боку осіб, причетних до галузі кібербезпеки, [10, 11] спонукали Сенатора Сполучених Штатів Рона Вайдена написати листа до Агентства національної безпеки та Міністерства оборони США з проханням провести аудит мобільного додатку "Voatz" [61].

1.5.2 Голосування в Москві у вересні 2019 року

Напередодні голосування код системи, що мала б використовуватись під час його проведення, було викладено у публічний доступ [33]. Метою публікації коду було заохочення фахівців-ентузіастів до тестування розробленого програмного забезпечення. Однак, через те, що жодної документації не було, на запит відгукнулось дуже мало людей. Окрім цього,

значну роль відіграло й те, що всі матеріали були російською, що значно зменшило кількість потенційних тестувальників.

З названих вище причин також не можна повністю розглянути особливості роботи системи. Оглянувши код, можна лише з упевненістю сказати, що система використовує Ethereum [29] та схему шифрування Ель-Гамаля [55].

Система для цього голосування була розроблена Департаментом інформаційних технологій міста Москва [45].

Для участі в інтернет-голосуванні необхідно було зареєструватись не пізніше як за 3 дні до початку голосування. А власне віддати голос можна було вже у сам день голосування одночасно з виборцями, що голосують класичним методом [45], на відміну від сценарію, за яким працює інтернет-голосування в Естонії.

Виборець, який голосував онлайн, міг віддати свій голос з будь-якого пристрою [45].

Усі голоси, що були віддані онлайн, дублювались на папері. На виборчих дільницях стояли принтери, які друкували бюлетені та персональні дані виборців, які віддали голос. Ці принтери вкидали бюлетені у спеціально відведені скриньки. Зміна голосу при застосуванні такого підходу є неможливою [45].

Також обіцяли забезпечити можливість приватної та публічної перевірки голосів. Приватна перевірка голосу можлива лише самим виборцем, який упевнюється, що його голос було подано до системи правильно і результат волевиявлення не змінено. Публічна ж перевірка дає змогу кожному перевірити чи певний голос у системі є коректним [45].

Зашифровані голоси у системі відправлялись до Ethereum блокчейну у вигляді транзакцій. Вони розшифровувались наприкінці голосування усередині смарт-контракту. Однак, у реалізації було знайдено вразливість. Тому було вирішено винести розшифрування за межі смарт-контракту [32].

Такі зміни були викликані тим, що для усунення вразливості необхідно було збільшити довжину ключа з 256 біт до 1024, а мова програмування Solidity, яка використовується у смарт-контрактах [63], має найбільше значення беззнакового цілочисельного типу $2^{256} - 1$. Через нестачу часу (адже до виборів залишалось декілька тижнів) було прийнято рішення винести розшифрування за межі смарт-контракту [32].

Таким чином, дослідники, що допомагали у покращенні системи шляхом її тестування, виділили дві головні проблеми цього підходу [32]:

- відсутність публічної специфікації;
- зміни, що були зроблені поспіхом напередодні голосування.

Висновки до розділу 1

Класичне паперове голосування справляється з поставленою задачею. Однак, для його проведення необхідна велика кількість людських та матеріальних ресурсів. Схеми, з допомогою яких можна здійснити неправомірний вплив на результат виборів добре відомі [5]. А аудит проведення голосування можуть здійснити не всі.

Scratch & Vote та PrÊt À Voter є методами, що базуються на паперових бюлетенях. Вони є значно прозорішими, порівняно з традиційним методом голосування, однак при цьому дещо ускладнюють виборцю задачу при голосуванні. При цьому використання даних методів голосування позбавляє від левової частки людського фактору, адже тут люди задіяні лише на етапі збору голосів та для аудиту кожного етапу. Окрім того, кількість людей, залучених до цього аудиту значно зростає, тому помилка однієї стає значно менш критичною.

Голосування в Естонії було інноваційним у 2000-х, однак зараз уже не виглядає надійним. Одним з недостатків є централізована природа системи. У разі, наприклад, зараження серверу передачі голосів, результати виборів будуть у руках зловмисника. Також, як і в паперовому голосуванні, присутній людський фактор, адже естонський метод голосування теж передбачає активне залучення людських ресурсів. Спостереження показали, що люди, відповідальні за різні завдання, під час проведення голосування допускають порушення протоколу, які можуть допомогти вчинити зловмисні дії у відношенні до голосування [60].

Методи голосування, що використовують блокчейн, хоч і привернули до себе увагу дослідників, але не поширені. Спільним недоліком московської системи та “Voatz” є недостатня прозорість через відсутність повної документації.

Варто розуміти, що введення електронного, а тим паче інтернет-голосування, хоч і позбавляє від ряду проблем, пов’язаних з класичним, та

підвищує загальну прозорість голосування, однак відкриває ряд нових викликів, які необхідно вирішити. Зокрема, забезпечення одночасно прозорості голосування, можливості для виборця бути впевненим у тому, що його голос було враховано відповідно до його волевиявлення, й неможливості доведення виборцем будь-кому змісту цього волевиявлення є доволі складною задачею.

РОЗДІЛ 2
МЕТОД ЦИФРОВОГО ГОЛОСУВАННЯ З ЧАСТКОВОЮ ПЕРЕВІРКОЮ

2.1 Блокчейн

Блокчейн був уперше представлений у якості бухгалтерської книги для біткоіну, у якій містяться записи про всі транзакції. Він використовується для захисту від подвійних витрат та модифікації записів про попередні транзакції.

Кожен вузол у блокчейні незалежно один від одного зберігає блокчейн, що містить лише блоки, які були затверджені цим вузлом. Коли декілька вузлів мають однакові блоки в ланцюгу, вони вважаються узгодженими. Правила затвердження, яким слідують ці вузли, називаються правилами узгодження. У цьому розділі описується багато правил, які використовуються у роботі біткоіну.

На рис. 2.1 зображена спрощена версія структури блокчейну. Копія кожної транзакції у блоці хешується, після чого хеші групуються попарно, хешуються та групуються знову допоки не залишиться один хеш, який є коренем дерева Меркла [13].

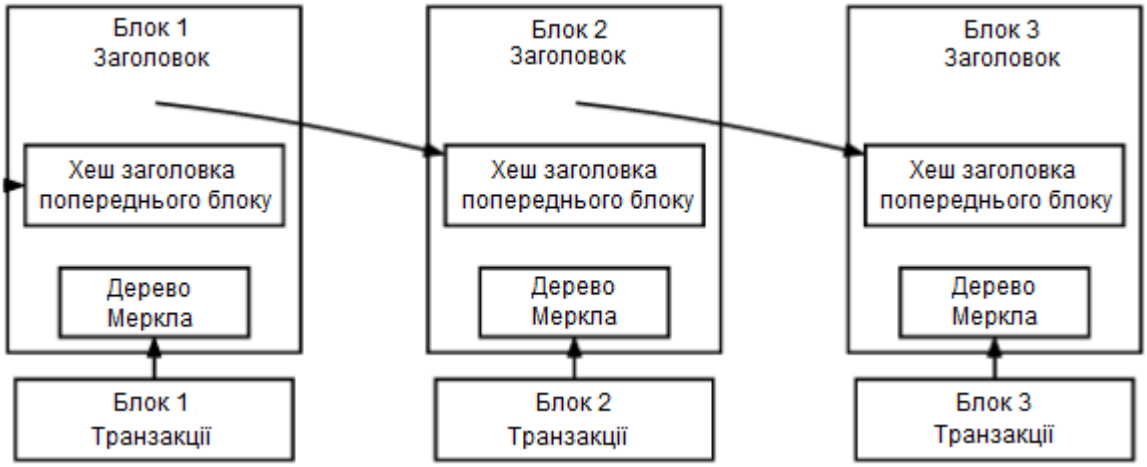

Рис. 2.1 Спрощена структура блокчейну

Цей корінь знаходиться у заголовку блоку. Кожен блок також містить хеш заголовка попереднього блоку, завдяки чому блоки об’єднуються у

ланцюг. Це забезпечує неможливість модифікації певного блоку без модифікації усіх, що йдуть за ним.

2.1.1 Транзакції

Транзакції також об’єднані у ланцюг. Користувачу, який здійснює транзакцію у біткоіні, здається що сатоші перекидаються між гаманцями, однак насправді обробляються транзакції. Кожна транзакція витрачає сатоші, що були перед цим отримані з іншої, тож входом однієї транзакції є вихід іншої [13].

Кожен власник (А) передає валюту підписуючи хеш попередньої транзакції та публічного ключа наступного власника (Б). Який, у свою чергу, з допомогою публічного ключа А перевіряє підпис. Схема утворення транзакцій зображена на рис. 2.2 [47].

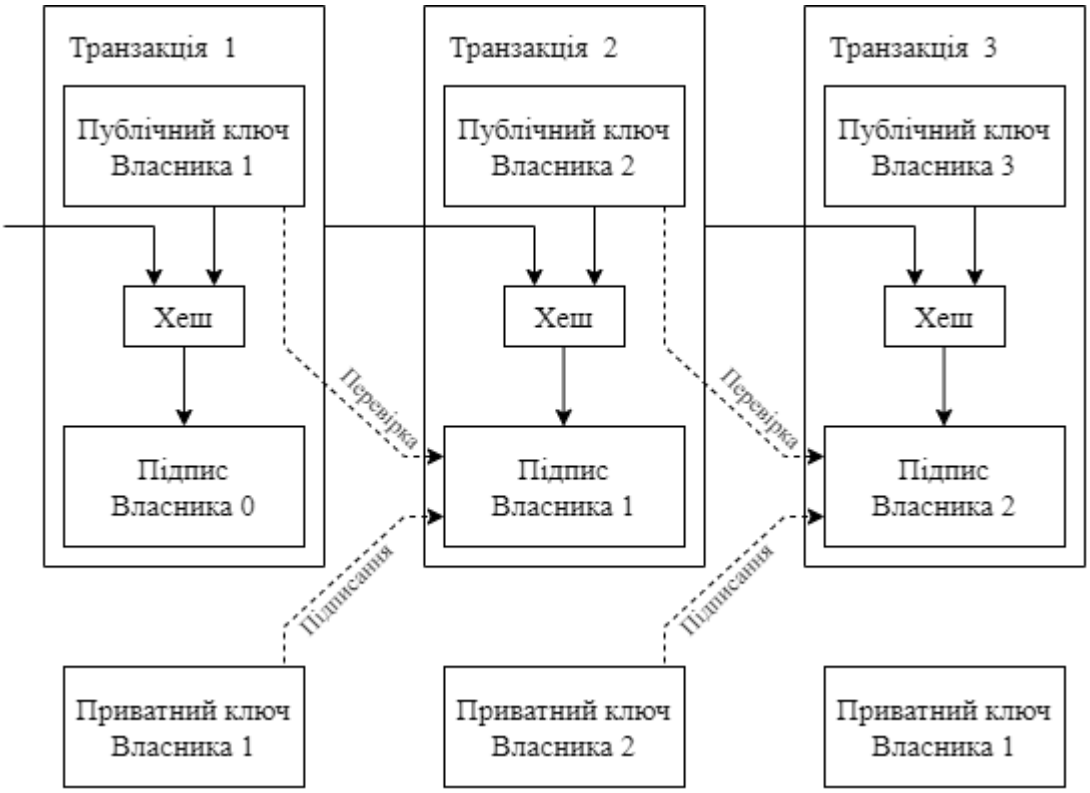

Рис. 2.2 Утворення транзакцій

Одна транзакція може мати декілька виходів, однак кожен вихід може бути використаний у якості входу лише один раз (рис. 2.3) [13].

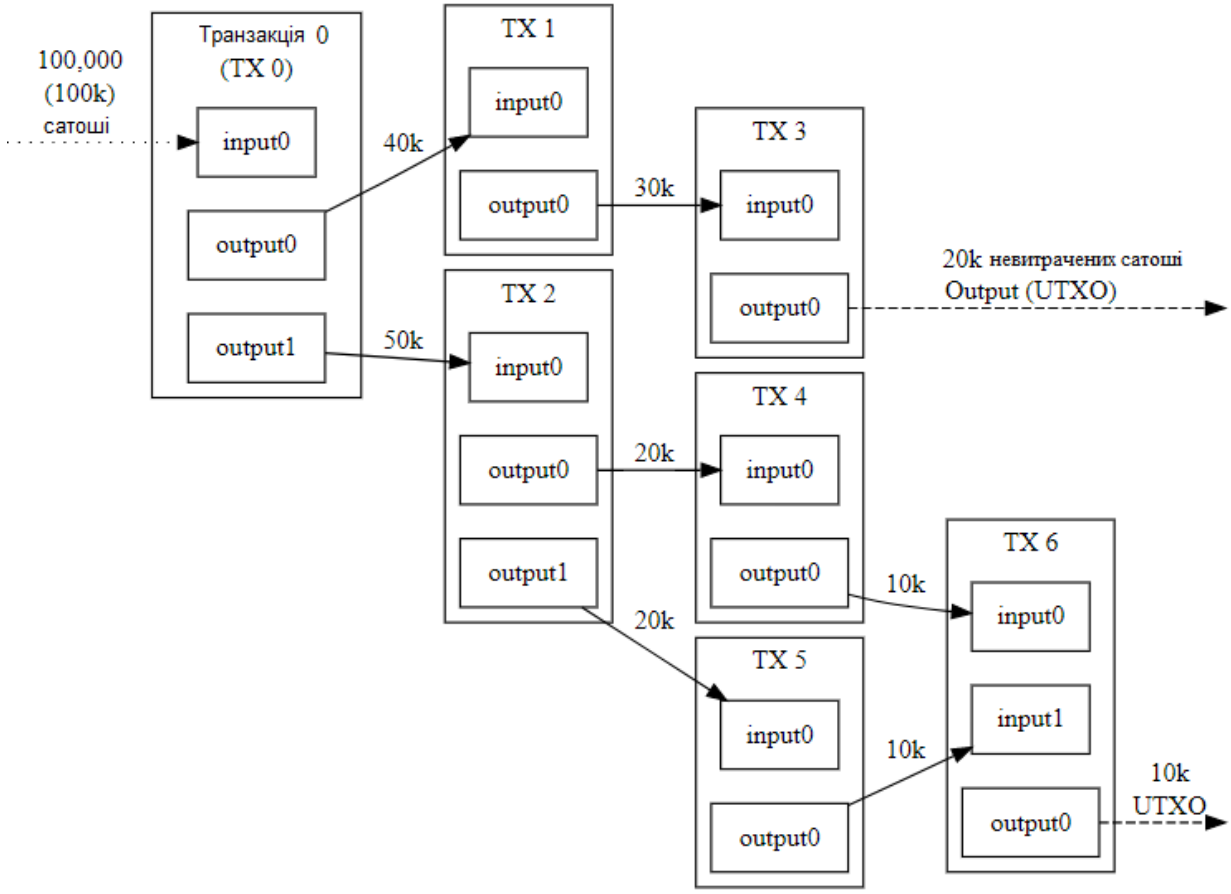

Рис. 2.3 Транзакції у біткоіні

Усі виходи прив’язані до ідентифікаторів транзакцій, які є їх хешами.

Так, як кожен вихід транзакції може бути витрачений лише раз, усі виходи можна поділити на витрачені та невитрачені. Якщо загальна сума входу перевищує загальну суму виходу, то транзакцію буде відхилено (за виключенням так званих coinbase-транзакцій, які будуть описані нижче). Окрім того, з кожної транзакції може бути стягнуто комісію, яку отримує майнер, що створив блок, який містить дану транзакцію. Наприклад, на рисунку вище кожна транзакція витрачає менше сатоші, ніж отримує, що означає, що 10000 сатоші з кожної транзакції було стягнуто в якості комісії.

Кожен блок повинен включати одну або більше транзакцій. Першою з них має бути coinbase-транзакція, що повинна зібрати та витратити нагороду майнера за створення блоку (включаючи усі комісії за транзакції у цьому блоці).

Невитрачений вихід (UTXO) coinbase-транзакції має особливу вимогу. Він не може бути витраченим принаймні 100 блоків. Це обмеження не дає майнеру витратити сатоші, що були отримані від створення блоку, який потрапив у побічну гілку блокчейну [16].

Блок може не включати в себе жодної транзакції, окрім coinbase. Утім, майнери практично завжди включають до блоку інші транзакції, щоб отримати з них комісії.

Усі транзакції, включаючи coinbase, закодовані у блоці в бінарному форматі. Цей бінарний формат хешується для створення ідентифікатора транзакції (TXID), з яких потім утворюється дерево Меркла. Якщо на якомусь етапі утворення дерева хеш не має пари, то він хешується сам з собою.

Кожна транзакція містить номер версії, що повідомляє пірам, який набір правил необхідно використовувати для її валідації. Це дозволяє розробникам створювати нові правила валідації таким чином, аби вони не конфліктували зі старими [16].

Кожен вихід транзакції має свій індекс, починаючи з 0, а також кількість сатоші, які він передає, та умовний скрипт публічного ключа (conditional pubkey script). Кожен, хто може задовольнити умову цього скрипту, може витратити сатоші, що належать цьому виходу [16].

Вхід використовує TXID та індекс для ідентифікації виходу. Він також містить скрипт підпису (signature script), який використовується для надання даних, що задовольняють умову скрипту публічного ключа [16].

Нехай користувач А збирається передати певну кількість сатоші користувачу Б. Для цього Б із свого публічного ключа формує хеш, який потім передає А у формі біткоін адреси, яка складається з номера версії, хешу публічного ключа та контрольної суми, закодованих з допомогою base58. Користувач А, у свою чергу, створює вихід транзакції, який містить скрипт публічного ключа, для підтвердження умови якого необхідний приватний ключ Б. Після цього А транслює транзакцію у мережу і вона додається до блокчейну.

Згодом, коли користувач Б захоче витратити сатоші, отримані від А, він створить нову транзакцію, вхід якої буде містити TXID попередньої транзакції, індекс виходу та скрипт підпису, який задовольняє умову попередньо згенерованого користувачем А скрипту публічного ключа. Усі описані дії проілюстровано на рис. 2.4 [16].

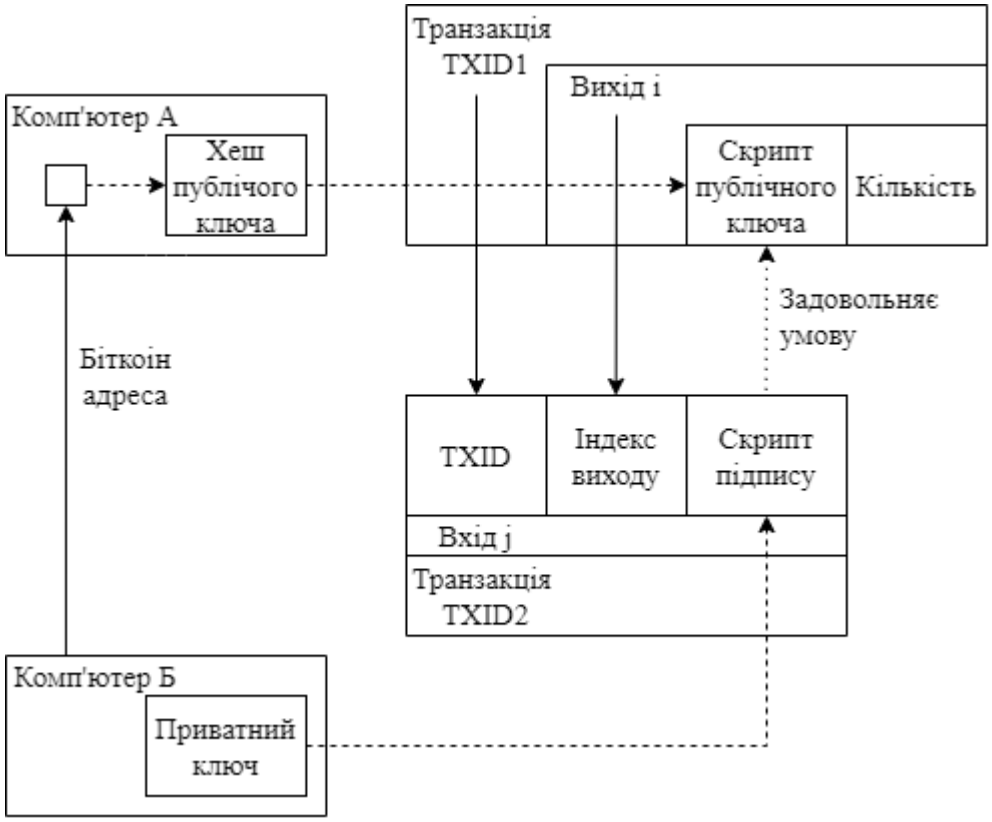

Рис. 2.4 Передача сатоші від користувача А до користувача Б

Таким чином, можна говорити про те, що блокчейн у реалізації біткоїна грає роль розподіленої бази даних транзакцій, що виконувались у системі, а поняття «гаманця» реалізується вже самим клієнтом.

2.1.2 Доказ виконання роботи

Роботу блокчейну забезпечують анонімні піри мережі, тож однією з вимог до біткоіну є те, що на створення одного блоку має бути виконано значну кількість роботи, аби забезпечити захист від модифікування блоків. Таким чином, неблагонадійні піри, що хочуть модифікувати блоки, повинні

виконати значно більше роботи, ніж добросовісні, що просто додають нові блоки до блокчейну.

Об’єднання блоків у ланцюг робить неможливою модифікацію транзакцій у певному блоці без зміни усіх наступних блоків. У результаті, ціна модифікації певного блоку зростає з кожним новим блоком, що доданий до блокчейну після нього, підсилюючи ефект доказу виконання роботи.

Доказ виконання роботи, який використовує біткоін, базується на базовому принципі хеш-перетворень. Хороший хеш-алгоритм перетворює дані у практично випадкове значення, що робить неможливим передбачити зміну хешу при зміні даних і навпаки.[47]

Для того, аби довести, що пір виконав певну роботу створюючи блок, він повинен створити хеш заголовку блоку, що не перевищує певного значення. Час, що в середньому необхідний для вирішення цієї задачі, лінійно залежить від значення: чим менше значення тим більше часу потрібно [13].

Кожні 2016 блоків блокчейн біткоіну перевіряє час, за який ці блоки були створені. Ідеальним значенням уважається близько двох тижнів. Якщо час значно відрізняється від ідеального, то складність завдання змінюється пропорційно до різниці між ідеальним та реальним часом створення блоків [13].

2.1.3 Висота блоку та розгалуження

Кожен майнер, що успішно виконав задачу хешування заголовку блоку, може додати блок до блокчейну. Блоки можна позначаити за їх висотою – кількістю блоків між вказаним та блоком 0. На блоці 2016, наприклад, може бути виконана перша корекція складності завдання для доказу виконання роботи [13].

Декілька блоків можуть мати однакову висоту, так як два або більше майнери можуть завершити блок приблизно в один час. Унаслідок цього у блокчейні можуть виникати розгалуження [13].

Коли виникає така ситуація, кожен вузол мережі незалежно визначає який блок він обирає у якості останнього у блокчейні [13].

З часом створюється наступний блок, який є продовженням гілки одного з блоків, які створились одночасно, що робить його гілку сильнішою. Усі піри завжди слідують гілці, яка має найбільшу складність, тому з часом (у межах кількох блоків) уся мережа знову синхронізується.

Так як декілька блоків можуть мати однакову висоту, то позначати їх за нею непрактично, тож для позначення блоку використовують його хеш, часто у зворотньому порядку, адже на перших бітах він матиме значну кількість нулів [13].

2.1.4 Контракти

Контрактами називають транзакції, що використовують блокчейн для підкріплення фінансових домовленостей [14].

Припустимо, наприклад, що користувач А хоче придбати щось у користувача Б. Тоді вони створюють простий контракт. Користувач А створює транзакцію, вихід якої містить скрипт, умову якого можна виконати лише якщо обидва користувачі підпишуть скрипт наступного входу. Такий контракт прекрасно працює поки все гаразд, однак що робити, якщо умови згоди все-таки було порушено. Тоді, очевидно, один із користувачів підписувати скрипт не буде і таким чином сатоші не зможе витратити ніхто. Тут користувачі А і Б звертаються до користувача В, який виступає третьою стороною. Вони створють ексроу контракт: користувач А створює вихід транзакції, який може бути витрачений якщо його підпишуть двоє з трьох сторін [14]. Отже, у разі виникнення суперечки між користувачами А та Б, користувач В може вирішити хто має отримати сатоші.

2.1.5 Валідація блокчейну

Біткоін передбачає два основних методи валідації блокчейну: метод усіх вузлів та метод SPV-клієнтів [15].

Метод усіх вузлів заключається в тому, що клієнт завантажує та перевіряє увесь блокчейн з блоку 0 до останнього, що йому відомий [15].

Очевидно, цей метод є громіздким, однак дієвим. Аби обманути клієнта і змусити його працювати з іншим блокчейном, необхідно створити блокчейн такої ж або більшої довжини, ніж основний [15].

Альтернативою ж є метод спрощеної перевірки платежів (Simplified Payment Verification, SPV). Клієнт завантажує лише заголовки блоків, які складають 80 байтів. За допомогою дерева Меркла у заголовку блоку можна визначити чи містить цей блок певну транзакцію. Дізнавшись у якому блоці ця транзакція, можна оцінити вартість атаки подвійної витрати виходу цієї транзакції, спираючись на глибину блоку (різниця між номером останнього блоку та даного) [15].

2.2 Криптосистема Пайє

Криптосистема Пайє [48] є асиметричним алгоритмом шифрування, який, подібно до RSA [56], використовує той факт, що факторизація числа, яке є добутком двох простих чисел – це дуже сколадна обчислювальна задача.

Нехай $n = pq$, де p і q є великими простими числами; позначимо через $\phi(n)$ фунцію Ейлера [44], а $\lambda(n)$ – фунцію Кармайкла [28]. Отримуємо $\phi(n) = (p-1)(q-1)$ і $\lambda(n) = \text{lcm}(p-1, q-1)$, де lcm – найменше спільне кратне. Врахуємо, що $\left|\mathbb{Z}_{n^2}^*\right| = \phi(n^2) = n\phi(n)$ і що для кожного $\omega \in \mathbb{Z}_{n^2}^*$:

$$\begin{cases} \omega^{\lambda(n)} = 1 \bmod n \\ \omega^{n\lambda(n)} = 1 \bmod n^2 \end{cases}$$

Визначення 1. Число z називається n-им залишком по модулю n^2 якщо існує таке число $y \in \mathbb{Z}_{n^2}^*$, що:

$$z = y^n \bmod n^2$$

Множина n-их залишків є мультиплікативною підгрупою $\mathbb{Z}_{n^2}^*$ порядку $\phi(n)$. Кожен залишок z має n коренів степеня n, з яких лише один є меншим за n.

Гіпотеза 2. Не існує поліноміального визначення часу, необхідного для виокремлення n-их залишків.

Нехай $g \in \mathbb{Z}_{n^2}^*$, визначимо $\mathcal{E}_g$ як функцію, що визначається як

$$\mathbb{Z}_n \times \mathbb{Z}_n^* \to \mathbb{Z}_{n^2}^*$$

$$(x, y) \to g^x \cdot y^n \bmod n^2$$

Лемма 3. Якщо g є кратним n, тоді функція $\mathcal{E}_g$ є бієктивною.

Позначимо $\mathcal{B}_\alpha \subset \mathbb{Z}_{n^2}^*$ множину елементів порядку $n\alpha$ та $\mathcal{B}$– їх об'єднання для $\alpha = 1 \ldots \lambda$.

Доказ. Так, як дві групи $\mathbb{Z}_n \times \mathbb{Z}_n^*$ та $\mathbb{Z}_{n^2}^*$ мають однакову кількість елементів $n\phi(n)$, необхідно довести, що функція $\mathcal{E}_g$ є ін'єктивною. Припустимо, що $g^{x_1} y_1^n = g^{x_2} y_2^n \bmod n^2$. Виходить $g^{x_2 - x_1} \cdot (y_2/y_1)^n = 1 \bmod n^2$, що передбачає $g^{\lambda(x_2 - x_1)} = 1 \bmod n^2$. Таким чином, $\lambda(x_2 - x_1)$ є кратним порядку g, а тому кратним n. Так, як $\gcd(\lambda, n) = 1$, $x_2 - x_1$ є кратним n. Як наслідок, $x_2 - x_1 = 0 \bmod n$ і $(y_2/y_1)^n = 1 \bmod n^2$, що приводить до єдиного розв'язку $y_2/y_1 = 1$ у $\mathbb{Z}_n^*$. Це означає, що $x_2 = x_1$, а $y_2 = y_1$. Таким чином, функція $\mathcal{E}_g$ є бієктивною.

Визначення 4: Припустимо, що $g \in \mathcal{B}$. Для $\omega \in \mathbb{Z}_{n^2}^*$, назвемо n-ий клас залишків ω у відношенні g унікальним значенням $x \in \mathbb{Z}_n$, для якого існує $y \in \mathbb{Z}_n^*$ такий, що

$$\mathcal{E}_g(x, y) = \omega$$

Позначимо клас ω як $[\omega]_g$.

Лемма 5. $[\omega]_g = 0$ якщо і тільки якщо ω є n-им залишком по модулю n^2. Більш того, $\forall \omega_1, \omega_2 \in \mathbb{Z}_{n^2}^*$ $[\omega_1 \omega_2]_g = [\omega_1]_g + [\omega_2]_g \bmod n$, тобто функція $\omega \to [\omega]_g$ є гомоморфною з $\left(\mathbb{Z}_{n^2}^*, \times\right)$ у $(\mathbb{Z}_n, +)$ для будь-якого $g \in \mathcal{B}$.

Проблема n-ого класу залишків з основою g (позначимо її $Class[n, g]$) визначається як проблема обчислення класової функції з основою g: для даного $\omega \in \mathbb{Z}_{n^2}^*$ обчислити $[\omega]_g$ з ω. Перш ніж визначати сладність $Class[n, g]$, зазначимо:

Лемма 6. $Class[n, g]$ самовідновлюється випадковим чином до $\omega \in \mathbb{Z}_{n^2}^*$.

Доказ. Можна легко випадковим чином перетворити $\omega \in \mathbb{Z}_{n^2}^*$ на $\omega' \in \mathbb{Z}_{n^2}^*$ з рівномірним розподілом за формулою $\omega' = \omega g^{\alpha} \beta^n \bmod n^2$, де α та β обрані випадковим чином за рівномірним розподілом з $\mathbb{Z}_n$ (подія, коли $\beta \in \mathbb{Z}_n^*$ дуже малоймовірна). Після підрахунку $[\omega']_g$ можемо визначити $[\omega]_g = [\omega']_g - \alpha \bmod n$.

Лемма 7. $Class[n, g]$ самовідновлюється випадковим чином до $g \in \mathcal{B}$, тобто

$$\forall g_1, g_2 \in \mathcal{B} \;\; Class[n, g_1] \equiv Class[n, g_2]$$

Доказ. Можна легко довести, що для кожного $\omega \in \mathbb{Z}_{n^2}^*$ та $g_1, g_2 \in \mathcal{B}$ маємо

$$[\omega]_{g_1} = [\omega]_{g_2} [g_2]_{g_1} \bmod n \tag{1}$$

з чого виходить $[g_1]_{g_2} = [g_2]_{g_1}^{-1} \bmod n$ і тому $[g_2]_{g_1}$ є зворотним за модулем n. Припустимо, що ми маємо передбачувач для $Class[n, g_1]$, тоді, передаючи у нього g_2 та ω, отримаємо відповідно $[g_2]_{g_1}$ та $[\omega]_{g_1}$, після чого зможемо обчислити

$$[\omega]_{g_2} = [\omega]_{g_1} [g_2]_{g_1}^{-1} \bmod n$$

Лемма 7 означаає, що складність обчислення $Class[n, g_1]$ не залежить від g. Це дозволяє розглядати її як обчислювальну задачу, яка залежить лише від n.

Визначення 8. Позначимо проблему класу складних залишків $Class[n]$, яка отримує $\omega \in \mathbb{Z}_{n^2}^*$ та $g \in \mathcal{B}$, а обчислює $[\omega]_g$.

Визначимо який зв'язок між $Class[n]$ та стандартними теоретико-числовими задачами.

Теорема 9. $Class[n] \Leftarrow Fact[n]$.

Перед доведенням теореми розглянемо множину

$$S_n = \{u < n^2 \mid u = 1 \bmod n\}$$

яка є мультиплікативною підмножиною цілих чисел за модулем n^2, серед яких функція

$$\forall u \in S_n \;\; L(u) = \frac{u-1}{n}$$

чітко визначена.

Лемма 10. Для кожного $\omega \in \mathbb{Z}_{n^2}^*$, $L\left(\omega^\lambda \bmod n^2\right) = \lambda[\omega]_{1+n} \bmod n$.

Доказ (Лемми 10). Так, як $1+n \in \mathcal{B}$, існує унікальна пара (a, b) у множині $\mathbb{Z}_n \times \mathbb{Z}_n^*$ така, що $\omega = (1+n)^a b^n \bmod n^2$. За визначенням $a = [\omega]_{1+n}$. Тоді

$$\omega^\lambda = (1+n)^{a\lambda} b^{n\lambda} = (1+n)^{a\lambda} = 1 + a\lambda n \bmod n^2$$

Доказ (Теореми 9). Так, як $[g]_{1+n} = [1+n]_g^{-1} \bmod n$ є зворотним, наслідком Лемми 10 є те, що $L\left(g^\lambda \bmod n^2\right)$ є зворотнім за модулем n. Тепер, факторизація n очевидно веде до знання λ. Тому, для будь-яких $\omega \in \mathbb{Z}_{n^2}^*$ та $g \in \mathcal{B}$, можемо обчислити

$$\frac{L\left(\omega^\lambda \bmod n^2\right)}{L(g^\lambda \bmod n^2)} = \frac{\lambda[\omega]_{1+n}}{\lambda[g]_{1+n}} = \frac{[\omega]_{1+n}}{[g]_{1+n}} = [\omega]_g \bmod n$$

за формулою 1.

Таким чином, маємо криптосестему:

Генеруємо $n = pq$ й обираємо випадковим чином основу $g \in \mathcal{B}$, що можна визначити перевіривши чи

$$\gcd\left(L\left(g^\lambda \bmod n^2\right), n\right) = 1$$

Тепер пара (n, g) є публічним ключем, у той час як пара (p, q) – приватним.

Шифрування:

$$c = g^m r^n \bmod n^2$$

де m – повідомлення, $m < n$; r – випадкове число, $r < n$.

Дешифрування:

$$m = \frac{L\left(c^{\lambda} \bmod n^2\right)}{L(g^{\lambda} \bmod n^2)} \bmod n$$

Варто звернути увагу на те, що випадкове число r не бере участі у дешифруванні.

2.3 Порогове шифрування

Порогова криптосистема складається із чотирьох компонентів [31]:

- Алгоритм генерації ключів у якості вхідних даних приймає параметр безпеки k, кількість серверів розшифрування l, пороговий параметр t та випадкову строку ω. Він генерує публічний ключ pk, набір приватних ключів $sk_1 \dots sk_l$ та набір ключів верифікації $vk_1 \dots vk_2$.
- Алгоритм шифрування приймає публічний ключ pk, випадкову строку ω та повідомлення m. Генерує шифротекст c.
- Розподілений алгоритм дешифрування приймає на вхід публічний ключ pk, індекс $i \in [1; l]$, приватний ключ sk_i та шифротекст c. Генерує частину c_i та доказ її правильності $proof_i$.
- Алгоритм об'єднання у якості вхідних даних приймає публічний ключ pk, шифротекст c, набір розшифрованих частин $c_1 \dots c_l$, набір ключів верифікації $vk_1 \dots vk_l$ та набір доказів $proof_i \dots proof_l$. На виході генерує повідомлення m.

Криптосистема Пайє може бути адаптована для алгоритму порогового шифрування. Чотири компоненти, описані вище, у пороговій криптосистемі мають такий вигляд:

- Алгоритм генерації ключів. Обираємо таке число n, що $n = pq$, де $p = 2p' + 1$, $q = 2q' + 1$ і при цьому p, q, p' і q' – прості числа, $\gcd\left(n, \phi(n)\right) = 1$. Вводимо $s = p'q'$. Нехай β – елемент $\mathbb{Z}_n^*$, обраний випадковим чином. Тоді випадковим чином обираємо $(a, b) \in \mathbb{Z}_n^* \times \mathbb{Z}_n^*$ і визначаємо $g = (1 + n)^a \cdot b^n \bmod n^2$. Приватний ключ $sk = \beta \cdot s$ розповсюджуємо за схемою Шаміра:

нехай $a_0 = \beta \cdot s$, випадковим чином обираємо t значень a_i у $\{0 \dots n \cdot s - 1\}$ і вводимо $f(X) = \sum_{i=0}^{t} a_i X_i$. Частина приватного ключа i-ого сервера $sk_i = f(i) \bmod ns$. Публічний ключ pk складається з g, n та значення $\theta = L\left(g^{s\beta}\right) = as\beta \bmod n$. Нехай vk – квадрат, що генеруються з циклічної групи квадратів $\mathbb{Z}_{n^2}^*$. Ключі верифікації отримуються за формулою $vk_i = vk^{\Delta sk_i} \bmod n^2$, де $\Delta = l!$.

- Алгоритм шифрування. Алгоритм шифрування нічим не відрізняється від звичайного:

$$c = g^m r^n \bmod n^2$$

- Розподілений алгоритм дешифрування. i-ий сервер обчислює свою розшифровану частину за формулою $c_i = c^{2\Delta sk_i}$, після чого формує доказ правильності розшифрування, який запевняє у тому, що $c^{4\Delta} \, mod \, n^2$ та $vk^{\Delta} \bmod n^2$ були піднесені до однієї й тієї ж степені sk_i, аби отримати c_i^2 та v_i відповідно.

- Алгоритм об'єднання. Нехай S – множина $t+1$ верифікованих розшифрованих частин, тоді

$$m = L\left(\prod_{j \in S} c_j^{2\mu_{0,j}^S} \, mod \, n^2\right) \cdot \frac{1}{4\Delta^2\theta} \bmod n$$

де $\mu_{0,j}^S = \Delta \cdot \prod_{j' \in S \setminus \{j\}} \frac{j'}{j'-j} \in \mathbb{Z}$

Зазначимо, що було обрано таке n, яке задовольняє $\gcd\left(n, \phi(n)\right) = 1$. Ця умова забезпечує те, що функція $f(a,b) = (1+n)^a \cdot b^n \bmod n^2$ є бієктивною з $\mathbb{Z}_n \cdot \mathbb{Z}_n^*$ у $\mathbb{Z}_{n^2}^*$.

Порядок g у $\mathbb{Z}_{n^2}^*$ – $n \cdot \alpha$, де α – порядок b у $\mathbb{Z}_n^*$. Більш того, можна помітити, що підгрупа квадратів у $\mathbb{Z}_{n^2}^*$ є циклічною і її порядок ns. Кількість генераторів у цій групі $\phi(ns)$, тож імовірність того, що випадково обраний квадрат із $\mathbb{Z}_{n^2}^*$ буде генератором приблизно $1 - 1/\sqrt{n}$, чого цілком достатньо. Тоді, ключі верифікації vk_i можна вважати свідками знання дискретного

логарифму vk_i у базі $vk^{\Delta} \bmod n^2$. Вони використовуються для генерації доказів того, що часткове дешифрування було проведено правильно.

Насамкінець, розглянемо підмножину S, що складається з $t+1$ правильно розшифрованих частин c_i. Обчислення $c^{4\Delta^2 s\beta}$ можна виконати з допомогою формули інтерполяції Лагранжа [46]:

$$\Delta f(0) = \Delta s\beta = \sum_{j \in S} \mu_{0,j}^{S} f(i) \bmod ns$$

тож

$$c^{4\Delta^2 s\beta} = \prod_{j \in S} c^{4\Delta s_j \mu_{0,j}^{S}} = \prod_{j \in S} c_j^{2\mu_{0,j}^{S}} \bmod n^2$$

Якщо c є шифротекстом повідомлення m, то

$$c^{4\Delta^2 s\beta} = g^{4\Delta^2 ms\beta} = (1+n)^{4\Delta^2 mas\beta} = 1 + 4\Delta^2 mas\beta \bmod n^2$$

Як наслідок, $L\left(\prod_{j \in S} c_j^{2\mu_{0,j}^{S}} \bmod n^2\right) = 4m\Delta^2 as\beta = m \cdot 4\Delta^2 \theta \bmod n$. Так, як θ – частина публічного ключа, отримуємо повідомлення m.

2.4 Мережа перемішування

Для забезпечення конфіденційності голосів виборців було використано концепцію мережі перемішування, яка полягає у тому, що деяка послідовність серверів по черзі бере партію вхідних повідомлень і перетворює їх на вихідні, змінюючи порядок. Жодна стороння особа не повинна мати змогу встановити відповідність між вхідними та вихідними повідомленнями [20]. У [20] усі повідомлення перед перемішуванням шифрувались з допомогою публічних ключів серверів, які будуть використовуватись для перемішування, у зворотному порядку.

Для забезпечення перевірки доброчесності кожного сервера було використано принцип випадкової часткової перевірки, що полягає у виборі частини повідомлень, для яких сервер повинен представити всі дані, з допомогою яких можна перевірити правильність утворення обраних пар.

Сервери у ланцюгу перемішування об’єднуються у пари, які не повинні надавати дані про одні й ті ж повідомлення у ланцюгу (рис. 2.5). Очевидно, що перетворення, які кожен сервер повинен розкрити, оголошуються після виконання обома серверами у парі всієї роботи.

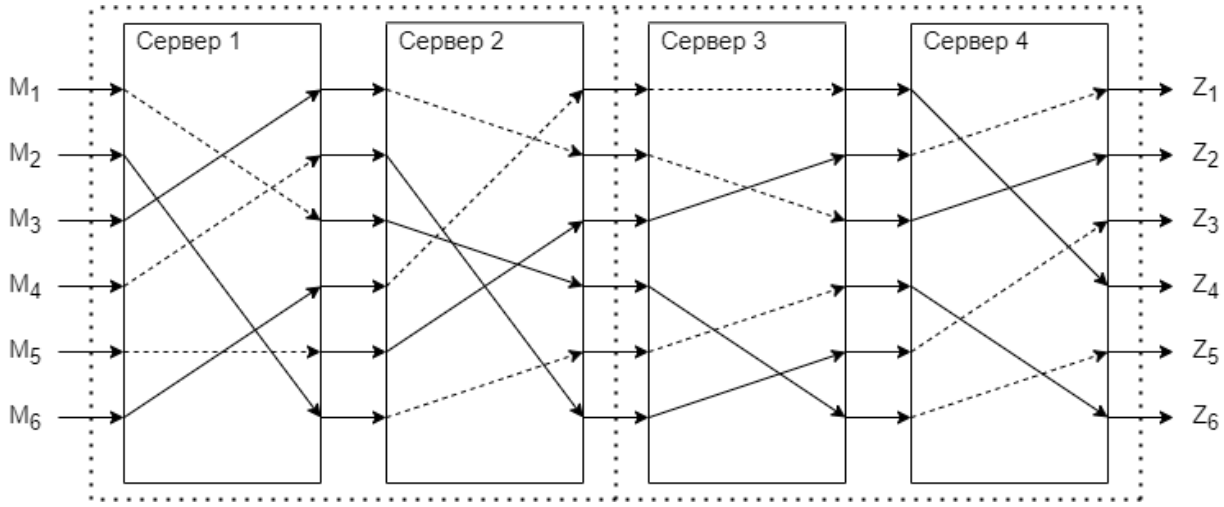

Рис. 2.5 Мережа перемішування на 6 повідомлень та 4 сервери

На рисунку присутні дві пари серверів: 1, 2 та 3, 4. Пунктиром позначені перетворення, що лишаються таємними. Сервери в парі не надають інформацію про перетворення одного й того ж повідомлення. Таким чином, імовірність встановити відповідність між повідомленнями знижується експоненційно з ростом кількості пар серверів.

2.5 Метод цифрового голосування з частковою перевіркою

Метод цифрового голосування, описаний у даній роботі, базується на методі цифрового голосування з частковою перевіркою, описаному в [1].

В організації голосування беруть участь декілька сторін, які представляють різні політичні напрями та інтереси. Кожна з них до проведення голосування надає певну кількість машин для проведення виборів і відповідає за їх обслуговування та коректну роботу. Усі машини реєструються та сертифікуються перед проведенням виборів. Вони не під’єднані до мережі Інтернет, але об’єднані за певною топологією в одну мережу, яка забезпечує роботу блокчейну, який буде виступати дошкою

оголошень під час голосування. Також у цій мережі присутні декілька машин, що мають вихід в Інтернет, однак у них немає прав на запис інформації до блокчейну, їх призначення – транслювати поточний стан блокчейну для широкої публіки. Усі N машин, що забезпечують роботу блокчейну складають множину серверів $\{SR_1 \dots SR_N\}$.

2.5.1 Цифровий бюлетень

Цифровий бюлетень складається з чотирьох полів:

- порядкового номера;
- ідентифікатора виборця;
- зашифрованого голосу;
- цифрового підпису виборця.

Голос має довжину l_v біт і складається з двох частин: R і V (рис 2.6). Частина R розміщена у старших бітах та має довжину $l_v - l_x$. Вона містить число, згенероване випадковим чином, яке необхідне для того, аби голос кожного виборця був унікальним. Частина V розміщена у молодших бітах голосу, вона має довжину l_x біт і призначена для кодування беспосередньо волевиявлення виборця.

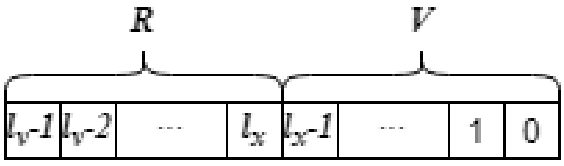

Рис. 2.6 Формат голосу у бюлетені

Частина голосу R генерується з трьох складових:

- випадкове число бюлетеня R_B – генерується на етапі генерації бюлетенів;
- випадкове число машини R_M – генерується машиною для голосування;
- випадкове число виборця R_V – генерується виборцем.

Для опису формування зашифрованого голосу введемо:

- $\mathrm{Enc}_{pk}(m)$ – функція шифрування повідомлення m з допомогою публічного ключа pk;
- $\mathrm{Dec}_{sk}(c)$ – функція дешифрування шифротексту c з допомогою ключа sk.

Напередодні голосування сервери $\{SR_1 \dots SR_N\}$ займаються генерацією бюлетенів.

Для генерації одного бюлетеня:

1) N_B серверів SR_i генерують по N'_B випадкових чисел R_{Bij}, таких що $R_{Bij} \in [2^{l_x}; (2^{l_v} - 1)/(3 \cdot N_B)]$;

2) кожен сервер SR_i шифрує кожне число R_{Bij} з допомогою публічного ключа pk та отримує масив $\left\{\mathrm{Enc}_{pk}(R_{Bi1}) \dots \mathrm{Enc}_{pk}\left(R_{BiN'_B}\right)\right\}$;

3) кожен сервер SR_i публікує свій масив у блокчейні голосування;

4) сервери $SR_j, j \in [1; N], j \neq i$ обирають із масиву кожного сервера $N'_B - 1$ значень $\mathrm{Enc}_{pk}\left(R_{Bij}\right)$ для аудиту;

5) сервери SR_j розшифровують обрані значення з допомогою порогового шифрування й публікують розшифровані R_{Bij} до блокчейну, що дає змогу з великою ймовірністю виявити будь-які злочинні дії з боку кожного SR_i, що згенерував масив для генерації бюлетеня;

6) із значень, які не було розшифровано $\left\{\mathrm{Enc}_{pk}(R_{B1}) \dots \mathrm{Enc}_{pk}\left(R_{BN_B}\right)\right\}$ формується

$$\mathrm{Enc}_{pk}(R_B) = \prod_{i=1}^{N_B} \mathrm{Enc}_{pk}(R_{Bi})$$

7) згенерований бюлетень публікується до блокчейну.

Таким чином, отримуємо

$$R_B = \sum_{i=1}^{N_B} R_{Bi}$$

Згенероване значення R_B є невідомим. Для того, аби дізнатись його, необхідно знати усі значення R_{Bi}, які були згенеровані різними серверами.

Виборець же напередодні голосування, з допомогою комп'ютера, використовуючи будь-яке програмне забезпечення, генерує свою частину випадкового числа та кодує своє волевиявлення. Оскільки процедура генерації є детально формально описаною, будь-хто може розробити програмне забезпечення, яке реалізує цю функцію. Тому виборець може використати програмне забезпечення від розробника, якому він довіряє, програмне забезпечення з відкритим кодом, або навіть власне програмне забезпечення для генерації голосу.

Генерація голосу виборця:

1) генерація випадкового числа R_V, такого, що $R_V \in [2^{l_x}; (2^{l_v} - 1)/3]$;
2) кодування волевиявлення $V \in [0; 2^{l_x} - 1]$;
3) обчислення $R_V + V$;
4) шифрування $\mathrm{Enc}_{pk}(R_V + V)$;
5) запис $\mathrm{Enc}_{pk}(R_V + V)$ у файл на цифровий носій.

Виборець може згенерувати й записати на носій скільки завгодно голосів, однак він повинен згенерувати певну кількість унікальних голосів h, аби машина для голосування дозволила йому використовувати даний носій для голосування.

Випадкове число R_M генерується машиною для голосування в момент голосування виборцем на виборчій дільниці.

2.5.2 Алгоритм голосування

У день голосування виборець приходить на виборчу дільницю. При собі він повинен мати:

- документи, що підтверджують особу;
- цифровий носій, на якому є h унікальних зашифрованих голосів та електронний цифровий підпис виборця.

Варто зазначити, що електронний цифровий підпис виборця генерується напередодні голосування і використовується тільки для участі в одному голосуванні, адже після використання приватний ключ перестає бути секретним. Цей підпис використовується для того, аби впевнитись що голос віддала саме та людина, яка отримала ідентифікатор.

На виборчій дільниці виборець проходить таку процедуру:

1) пред'являє документи, що підтверджують його особу та право голосувати;

2) відповідальні за виборчі списки члени комісії присвоюють виборцю ідентифікатор, фіксують це у списку виборців;

3) виборець отримує ідентифіктор та прямує до кабінки для голосування;

4) у кабінці виборець вставляє свій носій із зашифрованими голосами та електронним цифровим підписом;

5) машина здійснює перевірку безпеки носія, а також перевіряє чи на ньому є файли з h унікальними зашифрованими голосами, а також файл ЕЦП виборця;

6) виборець заповняє h бюлетенів з допомогою файлів з зашифрованими голосами на своєму носії; при цьому він не може використовувати один файл двічі;

7) машина обирає бюлетень із списку згенерованих $\mathrm{Enc}_{pk}(R_B)$, генерує R_M, шифрує його $\mathrm{Enc}_{pk}(R_M)$, зчитує вказаний виборцем файл $\mathrm{Enc}_{pk}(R_V + V)$;

8) машина обчислює

$$\mathrm{Enc}_{pk}(VT) = \mathrm{Enc}_{pk}(R_B) \cdot \mathrm{Enc}_{pk}(R_M) \cdot \mathrm{Enc}_{pk}(R_V + V)$$

9) виборець підписує заповнені бюлетені з допомогою свого ЕЦП;

10) машина вносить заповнені виборцем бюлетені до блокчейну;

11) виборець упевнюється в тому, що його бюлетені присутні в блокчейні;

12) виборець відправляє запит, підписуючи його своїм ЕЦП, який обирає з-поміж h згенерованих ним бюлетенів один, який й буде відображати його волевиявлення;

13) усі $h-1$ бюлетенів, окрім обраного виборцем, розшифровуються серверами SR_i з допомогою порогового шифрування та піддаються аудиту;

2.5.3 Підрахунок голосів

Після завершення голосування усі голоси в блокчейні можна без проблем відслідкувати, тому просто розшифровувати й підраховувати їх не можна.

Тому підрахунок голосів відбувається у три етапи:

1) перемішування,
2) розшифрування,
3) підрахунок.

На етапі перемішування N_{MN} серверів SR_i утворюють мережу перемішування [43] й перемішують N_{MB} бюлетенів. При цьому $N_{MN} \bmod 2 = 0$, так як сервери на цьому етапі поділяють на пари для аудиту.

Перемішування відбувається з допомогою техніки решифрування шифротексту, реалізацію якого дозволяє криптосистема Пайє.

Як було зазначено раніше, шифрування з допомогою криптосистеми Пайє здійснюється за формулою:

$$c = g^m r^n \bmod n^2$$

а дешифрування за:

$$m = \frac{L\left(c^\lambda \bmod n^2\right)}{L(g^\lambda \bmod n^2)} \bmod n$$

де c – шифротекст, m – повідомлення, n – публічний ключ, g – основа, така що $\gcd\left(L\left(g^\lambda \bmod n^2\right), n\right) = 1$, λ – приватний ключ, $L(x) = (x-1)/n$, а r – число, згенероване випадковим чином. Як бачимо, процедура дешифрування не має ніякої залежності від r.

Окрім того, як відомо, криптосистема Пайє має гомоморфні властивості, що дозволяє змінювати шифротекст c, не впливаючи при цьому на повідомлення m [48]:

$$\forall m \in \mathbb{Z}_n \;\; r \in \mathbb{N}$$

$$\mathrm{Dec}_{sk}\left(\mathrm{Enc}_{pk}(m) \cdot r^n \bmod n^2\right) = m$$

Таким чином, визначимо функцію $\mathrm{Rec}_{pk}(c, r)$ як операцію решифрування шифротексту c. Отримуємо:

$$\mathrm{Rec}_{pk}(c, r) = c \cdot r^n \bmod n^2$$

Саме ця властивість і дає змогу реалізувати мережу перемішування з допомогою криптосистеми Пайє. Таким чином, кожен сервер SR_i у мережі перемішування приймає на вхід множину зашифрованих голосів $\{C_1 \dots C_{N_{MB}}\}$ й формує множину шифротекстів за $\{C'_1 \dots C'_{N_{MB}}\}$:

$$C'_j = \mathrm{Rec}_{pk}(C_i, r_i)$$

де $i, j \in [1; N_{MB}]$.

Після цього, для аудиту етапу перемішування кожен сервер надає r_i та карту відношень j та i для кожного із обраних перетворень.

На етапі розшифровування $t + 1$ сервер, де t – порогове значення алгоритму шифрування, розшифровують кожен бюлетень й публікують розшифровані бюлетені до блокчейну.

Після цього частина R кожного голосу відкидається й голоси підраховуються.

Висновки до розділу 2

У даному розділі було спроектовано та детально формально описано метод цифрового голосування з частковою перевіркою та технології й алгоритми, що необхідні для його роботи.

Метод базується на технології блокчейн, яка виступає дошкою оголошень, гомоморфному пороговому шифруванні криптосистеми Пайє та мережах перемішування, яка необхідна для того, аби неможливо було встановити відповідність між виборцем та розшифрованим голосом.

Бюлетень у системі складається із чотирьох складових:

- R_B, що генерується групою серверів таким чином, що нікому не відомо його значення;
- R_M, що генерує машина для голосування;
- R_V, що генерує виборець напередодні голосування;
- V, що є закодованим волевиявленням виборця.

Машина у момент голосування генерує

$$\mathrm{Enc}_{pk}(VT) = \mathrm{Enc}_{pk}(R_B) \cdot \mathrm{Enc}_{pk}(R_M) \cdot \mathrm{Enc}_{pk}(R_V + V)$$

із чого, завдяки гомоморфній властивості криптостеми Пайє, випливає що

$$VT = R_B + R_M + R_V + V$$

Таким чином, число R складається з $R_B + R_M + R_V$, тому щоб дізнатись це число треба знати значення усіх випадкових чисел.

Використання такої генерації випадкового числа R разом з мережами перемішування з дуже високою ймовірністю гарантує те, що єдиним способом встановити відповідність між виборцем та розшифрованим голосом є перебір з-поміж близько $2^{l_v - l_x}$ варіантів, навіть якщо виборець надає достовірну інформацію про $R_V + V$.

Отже, можна говорити про те, що використання даного методу робить імовірність доведення виборцем того, як саме він проголосував, настільки малою, що нею можна знехтувати.

Схожою є й ситуація з перевіркою виборцем свого голосу. Шанс на те, що машина зможе підмінити голос виборця й підміна залишиться при цьому непоміченою становить $1/h$, що є доволі значною ймовірністю якщо обрати h розумної величини, з огляду на його призначення. Однак, імовірність виконати n_m таких підмін становить $1/h^{n_m}$, що говорить про те, що хоч виборець і не може бути впевненим у тому, що саме його голос було враховано правильно, зате він може бути впевненим у тому, що протягом голосування не було вчинено значного впливу на його результат.

Також варто зазначити, що використання для організації усього процесу голосування блокчейну значно підвищує його прозорість та стійкість до різноманітних фальсифікацій порівняно з іншими методами.

РОЗДІЛ 3
РЕАЛІЗАЦІЯ МЕТОДУ ЦИФРОВОГО ГОЛОСУВАННЯ З ЧАСТКОВОЮ ПЕРЕВІРКОЮ

3.1 Graphene

Graphene – фреймворк, з допмогою якого можна створити власний блокчейн [64]. Він включає всі компоненти, необхідні для створення блокчейн мережі. У той же час, цей фреймворк залишає можливість для розробників реалізувати власну логіку.

Graphene включає такі базові компоненти:

- базову реалізацію блокчейну зі сховищем даних;
- ефективний та продвинутий алгоритм синхронізації вузлів;
- завершену P2P мережу з HTTP API;
- фреймворк для додавання власного функціоналу;
- реалізацію різноманітних криптографічних функцій.

Використання Graphene дозволяє сконцентрувати увагу на логіці, що необхідна конкретно для блокчейну із даної роботи, не переймаючись про речі, які є загальними для всіх блокчейнів. Більш того, цей фреймворк є модулярним, що дозволяє змінювати будь-яку його частину під себе.

У порівнянні з іншими популярними реалізаціями блокчейну Graphene дозволяє організовувати зв’зок між вузлами швидше завдяки своєму алгоритму синхронізації вузлів.

Graphene використовує алгоритм, що називається «делегованим доказом частки володіння» (delegated proof of stake, який надалі у цій роботі буде згадуватись як DPoS). За цим алгоритом з-поміж вузлів блокчейну обираються довірені делегати (або свідки), які створюють блоки без необхідності додаткового майнингу. Свідки обираються у результаті голосування. Свідки переобираються раз у декілька блоків. Вони отримують винагороду за створення блоків. Кількість свідків та умови для того, аби стати одним із них,

можуть бути змінені вручну. Це дозволяє Graphene працювати ефективніше, ніж класичний підхід з PoW.

Як правило, свідок:

- неанонімний;
- є довіреним серед зацікавлених сторін;
- має достатньо обчислювальних потужностей.

DPoS організований так, що усі свідки є рівними, навіть якщо вони мають значно різну кількість голосів, що робить мережу невразливою до атаки 51%.

Смарт-контракти у Graphene прив’язані до самої мережі. Це покращує масштабованість та продуктивність. Хоч це й менш гнучкий підхід, однак логіку смарт-контрактів все ще можна змінити.

На базі Graphene побудовано декілька блокчейн мереж:

- BitShares [18],
- Steem [67],
- Peerplays [49],
- BitEthereum [17],
- Smoke [62],
- Scorum [59].

Кожна з цих мереж додає новий функціонал поверх ядра Graphene. Так, наприклад, Steem додає медіа токени до інших смарт активів.

Найбільш відомою реалізацією є BitShares. Він наслідує усі переваги Graphene, такі як швидкість та легка масштабованість, та додає багато нових особливостей, включаючи криптовалюту зі стабільною ціною (смарт-коїни), децентралізований обмін активами та автоматичні регулярні платежі.

Ці приклади показують наскільки гнучким є Graphene.

Graphene не має смарт-контрактів у їх звичайному значенні. Натомість уся логіка, яка необхідна розробнику, вбудована в мережу. Іншими словами, для того, щоб реалізувати смарт-контрат необхідно модифікувати вихідний код.

Фреймворк дозволяє легко додавати специфічні для розробника дії до мережі. Дія або операція у Graphene це просто функція у вихідному коді, до якої є декілька додаткових вимогих:

- Аргументи операцій мають бути типу struct, аби вони могли бути збереженими у блокчейні.
- Операція повинна мати функцію validate(), яка буде виконувати перевірку правильності переданих аргументів. Ця функція перевіряє чи всі аргументи мають сенс, правильно сформовані та влазять у відповідний тип даних. Так, наприклад, вона може перевіряти чи користувач не намагається передати криптовалюту самому собі, адже така операція не має сенсу.
- Операція повинна мати оцінювач. Оцінювач є мозком операції. Він складається з двох функцій:
    - do_evaluate(), яка перевіряє параметри, як і функція валідації. Однак ця функція проводить перевірку вищого рівня. Наприклад, чи має користувач достатньо токенів для проведення операції? Це не просто перевірка вводу, а повноцінна частина логіки мережі.
    - do_apply(), яка проводить необхідні перетворення. Вона, наприклад, може оновлювати баланс користувача, змінювати дані у блокчейні тощо.

Алгоритм виконання операції представлено на рис. 3.1.

Групи таких операцій формують смарт-контракти на платформі Graphene. Очевидно, оскільки ці «смарт-контракти» вбудовані безпосередньо у код мережі, немає легко способу змінити їх після того, як мережу було запущено. Однак, для цілей даної роботи змін у роботі мережі під час голосування не передбачається, а внести зміни між голосуваннями не буде проблематично. Тому, можна повною мірою скористатись такими перевагами фреймворку, як:

1) Швидкість – смарт-контракти скомпільовані у нативні виконувані інструкції так само, як і решта мережевого коду. Таким чином, смарт-контракт

викується настільки швидко, наскільки це можливо, без додаткового навантаження, спричиненого віртуальною машиною.

2) Безпека – неможливість додавання власних смарт-контрактів забезпечує захист від значного вектору можливих атак.

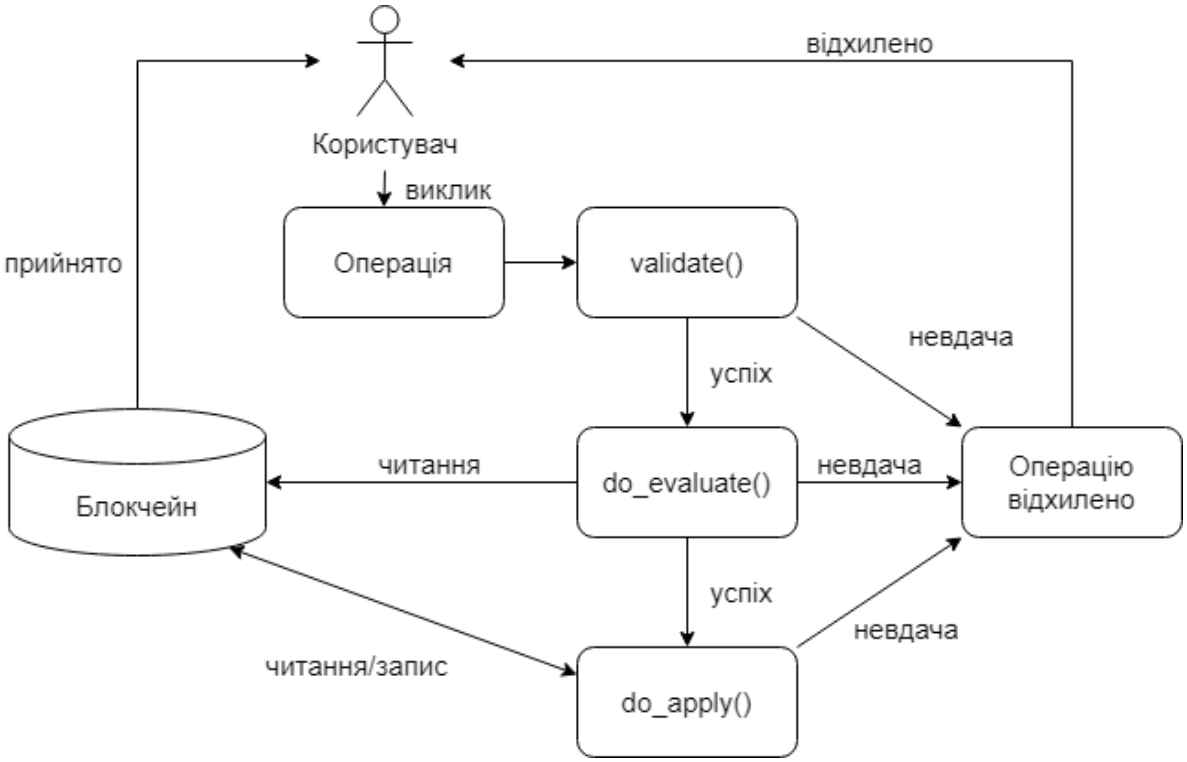

Рис. 3.1 Алгоритм операції в Graphene

Після того, як мережу було запущено, можна продовжити редагувати її відповідно до вимог розробника. Це є особливо корисним тоді, коли необхідно реалізувати основну масу особливостей одразу, а решту незначних змін робити по мірі появи вільного часу.

Є декілька способів розширити функціонал мережі:

- Додати більше опцій для доступу до операцій смарт-контрактів. Додавання команд до CLI може значно спростити задачу при тестуванні, а хороший графічний інтерфейс зацікавить та привабить нових користувачів.
- Додати більше продвинутих операцій до смарт-контрактів. Наприклад, можна додати функціонал, який з проходженням певного періоду часу очищає застарілі дані або оновлює стан аккаунтів.

- Додати підримку операцій між контрактами, наприклад користуватись контрактами на основі токенів власного дизайну, і, звичайно можна уточнити протоколи та покращити вже наявні функції.
- Змінити базовий функціонал фреймворку. Наприклад, якщо усі вузли у мережі можуть бути довіреними, то протокол синхронізації вузлів непотрібен.

Таким чином, цей фреймворк чудово підходить для реалізації методу цифрового голосування з частковою перевіркою на основі технології блокчейн.

3.2 Довга арифметика

Для реалізації описаного в дані роботі методу голосування використовується мова програмування C++ [40]. Ця мова програмування має сувору типізацію. Стандартна бібліотека STL не має реалізації довгої арифметики, яка без сумніву знадобиться для роботи з криптографічними алгоритмами.

Бібліотека, що називається GNU MP [71] є доволі відомою й призначена саме для роботи з великими числами. Однак, типи даних, описані в ній не підходять для роботи з Graphene, адже вони зберігають число у кучі, а на стеку – лише вказівники на область пам’яті, де воно зберігається та розмір. А для коректної роботи Graphene необхідно, аби весь тип даних повністю знаходився у struct.

Саме тому необхідно розробити власний тип даних, що зберігає числа довільної довжини та операції для роботи з ними.

Розроблений тип даних описано у файлах bignum.h та bignum.cpp.

Інтерфейс типу даних виглядає так:

```
template <size_t N>
struct bignum
{
private:
	uint8_t num[N];	// масив
```

```
public:
        bignum(unsigned int);       // створює bignum з unsigned int
        bignum(const bignum&);  // конструктор копіювання
        bignum(bignum&&);                   // конструктор переміщення

        bignum& operator=(const bignum&);    // оператор присвоєння копіювання
        bignum& operator=(bignum&&);                // оператор присвоєння
переміщення

        std::string to_string();       // функція представлення у вигляді строки
}
```

Також для цього типу даних описано операції додавання, віднімання, множення, ділення за модулем та піднесення до степеню з діленням за модулем.

3.3 Опис типів даних

Перед реалізацією необхідних операцій потрібно описати типи даних, які будуть брати в них участь і будуть записані до блокчейну.

Для зберігання будь-яких чисел, у чистому чи зашифрованому вигляді, використовується тип bignum, описаний вище.

Однак, окрім чисел, необхідно зберігати в блокчейні:

- бюлетені,
- масиви для генерації бюлетеня,
- згенеровані бюлетені,
- масиви, що проходять мережі перемішування,
- запити, з допомогою яких виборці обирають потрібний бюлетень,
- записи, з допомогою яких визначаються значення для аудиту,
- записи, що беруть участь у розшифруванні.

Бюлетені описані наступним чином:

```
template <size_t N1, size_t N2>
struct ballot : public graphene::db::abstract_object< ballot >
{
```

```
        unsigned int TYPE;              // тип запису
        unsigned int NUMBER;            // порядковий номер бюлетеня
        unsigned int VOTER_ID;          // ідентифікатор виборця
        bignum<N1> VOTE;                // зашифрований голос
        bignum<N2> VOTER_SIGN;          // цифровий підпис виборця
        unsigned int MACHINE_ID;        // ідентифікатор машини для голосування
        bignum<N2> MACHINE_SIGN;        // цифровий підпис машини для голосування
}
```

Масиви для генерації бюлетеня:

```
template <size_t N1, size_t N2, size_t N3>
struct proto_ballot_set_init : public graphene::db::abstract_object<
proto_ballot_set_init >
{
        unsigned int        TYPE;               // тип запису
        unsigned int        NUMBER;             // номер бюлетеню, що генерується
        unsigned int        MACHINT_ID;         // ідентифікатор машини
        bignum<N1>          R_b[N2];            // масив згенерованих R_b
        bignum<N3>          SERVER_SIGN;        // цифровий підпис сервера
}

template <size_t N1, size_t N2, size_t N3>
struct proto_ballot_set : public graphene::db::abstract_object< proto_ballot_set >
{
        unsigned int        TYPE;               // тип запису
        unsigned int        NUMBER;             // номер бюлетеню, що генерується
        unsigned int        MACHINE_ID;         // ідентифікатор машини
        unsigned int        SERVER_ID;          // ідентифікатор сервера
        bignum<N1>          R_b[N2];            // масив згенерованих R_b
        bignum<N3>          SERVER_SIGN;        // цифровий підпис сервера
}
```

Згенеровані бюлетені:

```
template <size_t N1, size_t N2>
struct proto_ballot : public graphene::db::abstract_object< proto_ballot >
{
        unsigned int TYPE;              // тип запису
        unsigned int NUMBER;            // номер згенерованого бюлетеню
        unsigned int MACHINE_ID;        // номер машини, якій належить бюлетень
        bignum<N1> R_b;                 // зашифроване випадкове число
        bignum<N2> MACHINE_SIGN;        // підпис машини для голосування
}
```

Масиви, що проходять мережі перемішування:

```
template <size_t N1, size_t N2, size_t N3 >
struct mix_net_set_initial : public graphene::db::abstract_object<
mix_net_set_initial>
{
        unsigned int        TYPE;           // тип запису
        unsigned int        NUMBER;         // номер мережі перемішування
        unsigned int        SERVER_ID;      // номер сервера
        ballot<N1, N2>      INPUT[N3];      // вхідний масив бюлетенів
        bignum<N1>          OUTPUT[N3];     // вихідний масив шифротекстів
        bignum<N2>          SERVER_SIGN;    // підпис сервера
}

template <size_t N1, size_t N2, size_t N3>
struct mix_net_set : public graphene::db::abstract_object< mix_net_set >
{
        unsigned int        TYPE;           // тип запису
        unsigned int        NUMBER;         // номер мережі перемішування
        unsigned int        SERVER_ID;      // номер серверу
        bignum<N1>          OUTPUT[N3];     // вихідний масив шифротекстів
        bignum<N2>          SERVER_SIGN;    // підпис сервера
}
```

Запит для вибору потрібного бюлетеня:

```
template <size_t N>
struct ballot_chooser : public graphene::db::abstract_object< ballot_chooser >
{
        unsigned int TYPE;              // тип запису
        unsigned int VOTER_ID;          // ідентифікатор виборця
        unsigned int BALLOT_NUMBER; // номер бюлетеню
        bignum<N>  VOTER_SIGN;          // підпис виборця
        bignum<N>  MACHINE_SIGN;        // підпис машини для голосування
}
```

Запис для голосування за випадкові значення при генерації бюлетеня:

```
template <size_t N1, size_t N2>
struct rb_vote : public graphene::db::abstract_object< rb_vote>
{
        unsigned int TYPE;              // тип запису
        unsigned int NUMBER;            // номер бюлетеня
        unsigned int MACHINE_ID;        // номер машини для голосування
        unsigned int SERVER_ID;         // номер сервера
        unsigned int R_b_numbers[N1]; // номери випадкових чисел у масиві
        bignum<N2> SERVER_SIGN;         // підпис сервера
}
```

Розподілене прийняття рішення:

```
template <size_t N>
struct decide : public graphene::db::abstract_object<music_contract_object>
{
        unsigned int        TYPE;                   // тип запису
        unsigned int        NUBMER:                 // номер мережі перемішування
        unsigned int        SERVER_ID;              // номер сервера, що голосує
        unsigned long long  shared_seed;            // значення, згенероване
                                                        сервером
        bignum<N>           SERVER_SIGN;            // підпис сервера
}
```

Голосування за перетворення, що піддаються аудиту у парі серверів, які взяли участь у мережі перемішування:

```
template <size_t N1, size_t N2>
struct mix_net_pair_reveal_vote : public
graphene::db::abstract_object<music_contract_object>
{
        unsigned int TYPE;              // тип запису
        unsigned int SERVER_ID;         // номер сервера, який голосує
        unsigned int SERVER1;           // номер першого сервера в парі
        unsigned int SERVER2;           // номер другого сервера в парі
        int          VOTES[N1];         // голоси за кожне перетворення {-1; 1}
        bignum<N2> SERVER_SIGN;         // підпис сервера, який голосує
}
```

Публікація інформації, необхідної для аудиту перетворень:

```
template <size_t N1, size_t N2, size_t N3>
struct mix_net_pair_reveal_set : public graphene::db::abstract_object
mix_net_pair_reveal_set>
{
        unsigned int TYPE;              // тип запису
        unsigned int SERVER_ID;         // номер сервера
        unsigned int INDICIES[N1];      // перемішування
        bignum<N3> R[N1];               // числа, що використані для решифрування
        bignum<N2> SERVER_SIGN;         // підпис сервера
}
```

Записи, що використовується для дешифрування:

```
template <size_t N1, size_t N2>
struct vote_decryption : public graphene::db::abstract_object< vote_decryption>
{
```

```
        unsigned int TYPE;              // тип запису
        unsigned int NUMBER1;           // номер машини або мережі перемішування
        unsigned int NUMBER2;           // номер бюлетеня або перестановки
        unsigned int SERVER_ID;         // нормер сервера, що дешифрує
        bignum<N1> C_i;                 // розшифрована частина
        bignum<N1> PROOF;               // доказ правильності дешифрування
        bignym<N2> SERVER_SIGN;         // підпис сервера
}
```

3.4 Додавання до фреймворку необхідних операцій

Реалізація методу цифрового голосування, що описаний у даній роботі, потребує додавання до фреймворку операцій, з допомогою яких можна виконати:

1) генерацію бюлетенів,
2) розподілення бюлетенів між виборчими дільницями,
3) запис до блокчейну бюлетенів, заповнених виборцем,
4) вибір виборцем дійсного бюлетеню,
5) мережу перемішування,
6) розшифрування шифротексту розподіленим ключем,
7) підрахунок голосів.

3.4.1 Генерація бюлетенів

Генерація бюлетенів проводиться перед голосуванням. Для генерації одного бюлетеня потрібно N_b вузлів, кожен з яких повинен згенерувати N_b2 зашифрованих випадкових чисел R_bi.

Для генерації бюлетеня машина для голосування, якій необхідний цей бюлетень створює запис proto_ballot_set у блокчейні. Усі інші сервери створюють свої записи з тим же номером бюлетеня доти, доки не назбирається принаймні N_b записів.

Машина починає генерацію бюлетеня та бере в ній участь з допомогою однієї й тієї ж операції generate_proto_ballot_set. Результати цих операцій відрізняються лише за полем TYPE.

- validate() у даному випадку перевіряє правильність написання номеру. Тобто, у випадку коли це ініціювання створення бюлетеню, перевіряється чи вказано унікальний номер для цієї машини, а у випадку генерації для іншої машини – чи запис про генерацію бюлетеня з таким номером у даної машини.
- do_evaluate() виконує перевірку того, чи наявно в блокчейні достатньо записів для бюлетеня, який генерується.
- do_apply() створює необхідний запис у блокчейні.

Усі сервери проводять моніторинг блокчейну і, коли набирається достатньо записів для певного бюлетеня, вони приймають рішення про те, які з R_bi будуть використані для генерації бюлетеня.

Тим часом, сервери, що брали участь у генерації бюлетеня, здійснюють моніторинг записів прийняття рішення з приводу чисел, які ці сервери згенерували випадковим чином та зашифрували для генерації цього бюлетеня. Коли кількість учасників досягає відмітки у N_b3, формує запис, який складається з усіх невибраних випадкових чисел, які згенерував сервер у розшифрованому вигляді.

- validate() перевіряє чи існує машина з таким номером та чи є у блокчейні записи про бюлетень з таким номером, а також перевіряє чи сходиться кількість згенерованих випадковим чином чисел, які відправляє сервер з тим, скільки було згенеровано до цього.
- do_evaluate() перевіряє чи відправлені числа дійсно відповідають тим, що були зашифровані раніше. Ця перевірка здійснюється шляхом виконання операції шифрування з використанням того ж публічного ключа над випадковим числом, що сервер представив у незашифрованому вигляді. Таким чином, виконуючи операцію C = encrypt(pk, R_bi_plain), перевіряємо чи C == R_bi_cypher. Якщо так, то представлене сервером число дійсно відповідає

тому шифротексту, який цей сервер надав раніше. Перевірка таким чином здійснюється тому що процедура дешифрування є розподіленою, а тому повільнішою. Через це використовувати її треба лише у випадках коли це дійсно необхідно.

- do_apply() виконує запис до блокчейну.

Завершальним етапом генерації бюлетеня є власне генерація бюлетеня з обраних з допомогою голосування R_bi. Для цього машина для голосування, що ініціювала створення бюлетеня виконує

```
bignum<N> R_b = 0;
for (unsigned int i = 0; i < N_b; ++i)
{
        R_b = R_b * R_bi[i];
}
```

Після генерації R_b машина створює запис proto_ballot у блокчейні.

- validate() перевіряє чи присутні усі необхідні записи у блокчейні. Так, наприклад, здійснюється перевірка того чи було взагалі ініційовано створення бюлетеня з таким нормером.
- do_evaluate() виконує перевірку того чи було віддано достатньо голосів, чи було згенеровано достатньо випадкових зашифрованих чисел та чи усі записи, у яких інші сервери надавали інформацію про свої згенеровані числа сформовані правильно й відповідають результатам голосування.
- do_apply() формує proto_ballot за фіксує його створення у блокчейні.

Таким чином, бюлетень з певним номером, вказаним при ініціюванні створення вважається згенерованим і готовим до використання при голосуванні.

Окрім того, так як створення бюлетеня ініціює машина для голосування, якій потім і належатиме бюлетень, даний алгоритм генерації вирішує також і проблему розподілення бюлетенів між виборчими дільницями й машинами для голосування.

3.4.2 Забезпечення голосування

Під час проведення голосування виборець у кабінці для голосування заповнює свій бюлетень за допомогою цифрового носія, який містить набір зашифрованих голосів, необхідних для голосування та електронний цифровий підпис виборця, який згенерований спеціально для використання на поточному голосуванні.

У момент коли виборець вставляє свій носій до порту машини для голосування, машина, працюючи з директорією, що вказана у її конфізі, спершу перевіряє носій на наявність будь-яких файлів, що не підходять по формату для проведення голосування. У разі знаходження таких файлів машина демонтує носій з файлової системи та виводить повідмлення про помилку. Якщо ж перевірку пройдено успішо, то машина переходить до наступного кроку.

На наступному кроці виконується перевірка файлів. Тобто, машина перевіряє, чи наявна на носії встановлена кількість унікальних голосів, необхідних для голосування, а також перевіряє чи наявний на носії електронний цифровий підпис виборця.

Після проходження усіх необхідних перевірок виборець нарешті може здійснити голосування. Для того, аби надати виборцю таку можливисть, машина завантажує з блокчейну певну кількість бюлетенів та виводить графічний інтерфейс, який виводить інформацію про обрані бюлетені, а також надає можливість встановити відповідність між бюлетенями та файлами, у яких зашифровані голоси виборця.

Обравши файли з голосами, виборець виконує цифровий підпис згенерованих бюлетенів.

Після заповнення усіх бюлетенів машина формує записи ballot для кожного бюлетеня та публікує їх у блокчейні.

Для цього виконується операція push_ballot, яка має таку структуру:

- validate() виконує перевірку коректності усіх параметрів та відповідність номерів бюлетенів до та після голосування.
- do_evaluate() перевіряє чи виборець з таким ідентифікатором ще не голосував, а також перевіряє чи виборець з таким ідентифіктором існує та зареєстрований у системі й має право голосу на даній виборчій дільниці.
- do_apply() записує згенеровані бюлетені до блокчейну.

Заповнивши та підписавши всі необхідні бюлетені виборець очікує доки вони з’являться у блокчейні, аби бути впевненим у тому, що його голос буде зараховано.

Упевнившись у наявності своїх бюлетенів у блокчейні, виборець використовує графічний інтерфейс, що був представлений машиною одразу після відправлення бюлетенів, для того, аби обрати бюлетень з яким номером він хоче використати у якості свого справжнього голосу.

При цьому машина генерує запис ballot_chooser, який містить усю необхідну інформацію.

- validate() виконує перевірку того чи належить вказаний бюлетень даному виборцю.
- do_evaluate() перевіряє чи не було до цього уже виконано запис про обирання бюлетеня виборця з таким ідентифікатором.
- do_apply() формує ballot_chooser та записує його до вмісту блокчейну.

Усі сервери під час проведення голосування неперервно моніторять блокчейн на наявність бюлетенів, які необхідно розшифрувати. Тому, у момент, коли сервер помічає запис ballot_chooser, він завантажує бюлетені, які необхідно розшифрувати, проводить операцію дешифрування, використовуючи свою частину ключа, генерує доказ того, що розшифрування було проведено правильно та генерує запис, що повідомляє інших учасників блокчейну про те, що він виконав часткове дешифрування бюлетеню, призначеного для аудиту.

- validate() перевіряє чи існує вказаний бюлетень, а також чи вказаний бюлетень дійсно призчений для аудиту, а виборець у якості свого волевиявлення обрав інший.
- do_evaluate() перевіряє чи дійсний доказ того, що розшифрування пройшло правильно.
- do_apply() робить відповідний запис vote_decryption у блокчейні.

Сервери також моніторять мережу й на наявність достатньої кількості розшифрованих частин, аби розшифрувати бюлетень. Виявивши, що у мережі наявна необхідна кількість частин, сервери приймають рішення про те, який сервер буде займатись комбінуванням розшифрованих частин.

Сервер, що був обраний у результаті голосування після того, як було набрано необхідну кількість голосів, виконує комбінування розшифрованих частин відповідно до алгоритму порогового шифрування.

- validate() перевіряє чи існує вказаний бюлетень, а також чи вказаний бюлетень дійсно призчений для аудиту, а виборець у якості свого волевиявлення обрав інший.
- do_evaluate() перевіряє чи дійсно даний сервер був обраний для публікації результату дешифрування, а також чи було для цього віддано достатньо голосів.
- do_apply() публікує розшифрований бюлетень разом з інформацією, необхідною для його ідентифікації, до блокчейну.

Таким чином, виконується розшифрування бюлетеню, при цьому в блокчейні записано доказ правильності кожного часткового розшифрування, а результат комбінування розшифрованих частин кожен спостерігач може виконати самостійно виконавши алгоритм.

3.4.3 Прийняття рішень

Для того, аби у системі був порядок і аби сервери могли приймати спільні рішення, необхідний алгоритм прийняття рішень.

Найбільшою складністю при пошуку такого алгоритму є те, що треба позбавити кожен сервер можливості вплинути на кінцеве рішення таким чином, щоб отримати для себе певну вигоду. Тобто, кожен сервер повинен впливати на прийняття рішення, однак не повинен знати як саме він на нього впливає.

Для цього кожен сервер, коли бачить, що у блокчейн мережі необхідно прийняти рішення, генерує стурктуру даних decide, у якій записує своє число типу unsigned long long, яке буде його вкладом до прийняття рішення.

Коли набирається достатня кількість серверів, що взяли участь у прийнятті рішення, сервери починають обчислювати це рішення.

Для обчислення рішення усі числа типу unsigned long long, згенеровані серверами, які були поміщені у структуру даних decide, сумуються. Після цього їх сума хешується.

Отриманий хеш використовується у якості seed для генератора псевдовипадкових чисел при формуванні рішення.

Найпершим при формуванні будь-якого рішення вирішується який сервер повинен сформувати це рішення та зробити відповідний запис. Для цього використовується цей самий seed.

Після цього сервер, який у результаті має опублікувати рішення, за заздалегідь встановленим алгоритмом генерує рішення, у створенні якого взяли участь сервери.

- validate() первіряє правильність усіх даних, що були згенеровані, їх формат та розмір.

- do_evaluate() перевіряє чи достатня кількість серверів узяла участь у прийнятті рішення; також перевіряє чи дійсно усі згенеровані значення були отримані з допомогою використання правильного seed.
- do_apply() фіксує прийняте серверами рішення у блокчейні.

Таким чином, жоден сервер, за розумний час не може обчислити значення, яке він має надати для участі в алгоритмі прийняття рішень, для того, аби якось свідомо вплинути на це рішення.

3.4.4 Мережа перемішування

Після завершення голосування голоси необхідно перемішати. Для цього виконується алгоритм, що має назву «мережа перемішування». Для забезпечення роботи цього алгоритму необхідна послідовна обробка певною множиною серверів певної множини бюлетенів. Таким чином, необхідно встановити порядок, у якому N_mn серверів будуть перемішувати N_mv голосів. Для цього спочатку виконується голосування за сервер, який відбере множину голосів та виконає перше перетворення, після чого виконується голосування за сервер, який буде виконувати друге перетворення й так далі. У цей же час відбуваються голосування за сервери, які вібирають голоси та виконують перетворення для інших мереж перетворення.

Усі сервери за алгоритмом прийняття рішень вирішують який сервер буде наступним у мережі перемішування.

По завершенню прийняття рішення сервер, який було обрано, у випадку, якщо він перший у мережі, виконує відбір необхідної кількості бюлетенів для мережі. Якщо ж сервер у мережі не є першим, він просто виконує перемішування за алгортимом решифрування. При цьому сервер повинен запам’ятати дані, які дозволять відтворити перетворення пізніше, адже йому потрібно буде надати частину з них для аудиту.

- validate() перевіряє чи сервер дійсно було обрано у якості наступного у мережі перемішування та що за нього було віддано достатньо голосів.

- do_evaluate() перевіряє чи даний сервер не був задіяний у даній мережі перемішування раніше.
- do_apply() записує результат перемішування до блокчейну.

Після проходження множиною голосів достатньої кількості серверів усі сервери в мережі розуміють, що перемішування завершено. Вони групують сервери, що брали участь у роботі мережі, по два, після чого голосують за перемішування, інформацію для здійснення яких мають надати ці сервери. При цьому сервери у парі не можуть надавати інформацію про перетворення, що відповідають за одні й ті ж голоси. Таким чином, перший сервер у парі надає інформацію про одну половину перетворень, а другий – про іншу.

Після цього сервери приймають рішення про те, які з перетворень кожна пара серверів у кожній мережі перемішування повинна представити.

Сервери, що брали участь у мережі перемішування, моніторять мережу на предмет прийняття рішення з приводу їх перетвореньь. Коли набирається достатня кількість серверів для прийняття рішення, сервер підраховує голоси, обчислюючи значення, що відповідає кожному перетворенню. Тепер, сервер, що був першим у парі, надає інформацію про значення, які після обчислення результату голосування мають менші значення, а той, що був другим – ті, що мають більші значення. За наявності однакових значень, які вносять двоякість у результат розподілення за таким алгоритмом, беруться до уваги й порівнюються номери перетворень, які не можуть бути однаковими, а тому будь-яку неясність можна вирішити.

Визначивши які перетворення мають бути виявленими, сервер публікує інформацію, що необхідна для відновлення обраної половини його перетворень.

- validate() перевіряє чи операцію виконує дійсно той сервер, який має її виконувати, та перевіряє кількість елементів у масиві, яка повинна відповідати кількості бюлетенів, залучених до мережі перемішування.

- do_evaluate() перевіряє чи у масиві наявна інформація про всі перетворення, інформацію про які мав надати сервер, а також що у нього немає інформації про перетворення, інформацію про які сервер надавати не мав.
- do_apply() здійснює запис mix_net_pair_reveal_set з інформацією про перетворення до блокчейну.

Таким чином, бюлетені стають перемішаними й при цьому кожен аудитор може впевнитись у тому, що з високою ймовірністю процес пройшов чесно.

3.4.5 Розшифрування та підрахунок голосів

Після перемішування голоси необхідно розшифрувати та підрахувати. Для цього сервери проводять моніторинг мережі на предмет того, чи усі голоси було включено до мереж перемішування та чи усі мережі перемішування було завершено. Окрім того, сервер упевнюється в тому, що усі мережі пройшли аудит.

Коли всі ознаки закінчення перемішування голосів у блокчейні наявні, серевери починають розшифровувати перемішані голоси. Для цього сервери обирають випадковим чином голоси з-поміж тих, які потребують розшифрування, після чого здійснюють часткове розшифрування за алгоритмом та розміщують результат у блокчейні разом з доказом того, що розшифрування було проведено правильно.

- validate() здійснює перевірку того, що усі голоси було включено до мережі перемішування та що усі мережі перемішування було завершено; також перевіряє чи усі мережі пройшли перевірку.
- do_evaluate() перевіряє чи потребує вказаний голос часткового розшифрування, тобто, чи не було уже згенеровано для нього достатньо часткових розшифрувань.
- do_apply() записує результат часткового розшифрування до блокчейну.

Коли для якогось із голосів набирається потрібна кількість розшифрованих частин, починається прийняття рішення з приводу сервера, який буде здійснювати комбінування розшифрованих частин.

Після прийняття рішення обраний сервер проводить комбінування розшифрованих частин за алгоритмом.

- validate() перевіряє чи існує вказаний голос.
- do_evaluate() перевіряє чи дійсно даний сервер був обраний для публікації результату дешифрування, а також чи було для цього віддано достатньо голосів.
- do_apply() публікує розшифрований голос до блокчейну.

Таким чином, у блокчейні з часом розміщаються усі розшифровані голоси, після чого будь-хто може виконати підрахунок голосів.

3.5 Реалізація криптосистеми Пайє

Описаний у даній роботі метод голосування з частковою перевіркою, що працює на основі технології блокчейн, використовує криптосистему Пайє для шифрування, що означає, що необхідно реалізувати бібліотеку, яка виконує шифрування та розподілене дешифрування на мові програмування C++.

Усі операції, що стосуються криптосистеми Пайє, описано у класі Paillier, який описано у файлах paillier.cpp та paillier.h.

Інтерфейс розробленого класу виглядає таким чином:

```
template<size_t N>
class Paillier
{
public:
	Paillier() = default;  // конструктор за замовчуванням

	/*
	Функція, що реалізує шифрування, приймає у якості параметрів публічний ключ publick_key, текст plaintext та випадкове число random
```

```
        Повертає зашифрований публічним ключем текст, для шифрування якого
        було використано вказане випадкове число
        */
        bignum<2*N> encrypt(bignum<N> public_key, bignum<N> plaintext,
                bignum<N> random);


        /*
        Функція, що виконує решифрування шифротексту cyphertext з
        використанням публічного ключа public_key та випадкового числа random
        Повертає решифрований шифротекст
        */
        bignum<2*N> re_encrypt(bignum<N> public_key, bignum<2*N> cyphertext,
                bignum<N> random);

        /*
        Функція, що виконує чаткове дешифрування шифротексту cyphertext з
        допомогою розподіленого приватного ключа shared_secret_key
        Повертає часткове розшифрування повідомлення
        */
        bignum<2*N> shared_decrypt(bignum<N> shared_secret_key,
                bignum<2*N> cyphertext);


        /*
        Функція, що генерує доказ того, що шифротекст shared_cypher є частковим
        розшифруванням cyphertext з допомогою ключа secret_key
        Повертає доказ правльності дешифрування
        */
        bignum<N> make_shared_proof(bignum<N> shared_secret_key,
                bignum<2*N> cyphertext, bignum<2*N> shared_cypher);

        /*
        Функція, що перевіряє доказ дешифрування
        */
        bool check_shared_proof(bignum<2*N> cyphertext,
                bignum<2*N> shared_cypher, bignum<N> shared_proof);

        /*
        Функція, що об'єднує часткові дешифрування
        Повертає розшифрований текст
        */
        bignum<N> combine_decryption(bignum<N>* shared_cyphers);
}
```

Як бачимо, у класі реалізовано 6 функцій, які виконують шифрування, розшифрування, генерацію доказу розшифрування, перевірку доказу розшифрування та комбінування часткових розшифрувань.

Клас не виконує операції генерації ключів, адже використання методу голосування, що описаний у даній роботі передбачає те, що всі ключі вже згенеровано та всі учасники блокчейну мають свої ключі та знають публічний ключ.

3.6 Цифровий підпис

Для реалізації методу голосування також потрібна технологія цифрового підпису. У дані роботі для забезпечення цифрового підпису скористаємось також криптосистемою Пайє. Для цього до класу Paillier, описаного раніше, було додано функцію

```
bignum<N> encrypt(bignum<N> secret_key, bignum<2*N> cyphertext);
```

яка виконує розшифрування шифротексту cyphertext з допомогою секретного ключа secret_key.

Після цього, було описано клас DigitalSignature, який розміщений у фалах digitalsignature.cpp і digitalsignature.h.

```
template<size_t T>
class DigitalSignature
{
private:
	Paillier paillier;
public:
	DigitalSignature(); // конструктор

	/*
	Функція, призначена для хешування даних, які треба підписати
	*/
	bignum<N> sha256(bignum<32>* data, size_t length);

	/*
```

```
        Функція, призначена для підпису даних data довжиною length з допомогою
        приватного ключа secret_key
        */
        bignum<N> sign(bignum<N> secret_key, bignum<32>* data, size_t length);

        /*
        Функція, з допомогою якої можна перевірити підпис даних data довжини
        length з допомогою публічного ключа public_key
        */
        bool check_signature(bignum<N> public_key, bignum<32>* data,
                size_t length);

}
```

Висновки до розділу 3

У цьому розділі описано реалізацію методу цифрового голосування з частковою перевіркою на основі технології блокчейн.

Для реалізації блокчейну було обрано фреймворк Graphene, написаний на мові програмування C++. Перевагами цього фреймворку є його швидкодія та відкритий вихідний код, що дає змогу змінювати його за потреби.

Окрім цього, Graphene має продвинутий алгоритм синхронізації вузлів DPoS, який чудово підходить для реалізації задач, подібних, до тієї, що вирішується в даній роботі.

Також на базі фреймворку Graphene було описано усі необхідні для роботи типи даних та імплементовано усі необхідні операції, зокрема

- операцію генерації бюлетеня;
- операцію голосування за сервер;
- операцію голосування за значення, згенероване сервером;
- алгоритм роботи мережі перемішування;
- операцію розшифрування голосів.

Було реалізовано операції, які необхідні для використання криптосистеми Пайє.

Також для забезпечення роботи механізму цифрового підпису було реалізовно клас, що виконує усі необхідні операції для виконання та перевірки електронного цифрового підпису.

Таким чином, було створено систему, яка реалізує метод цифрового голосування з частковою перевіркою на основі технології блокчейн.

РОЗДІЛ 4
АНАЛІЗ ОТРИМАНИХ РЕЗУЛЬТАТІВ

4.1 Опис параметрів системи

Система, що була розроблена у розділі 3, має низку ключових параметрів, які впливають на її надійність та стійкість до різноманітних загроз. Ці параметри включають:

- N – кількість вузлів у мережі блокчейн;
- N_B – кількість серверів, необхідних для генерації бюлетеня;
- N_B' – кількість варіантів R_{Bi}, які генерує один сервер;
- l_v – розмір голосу;
- l_x – розмір частини голосу V, що призначена для волевиявлення;
- l_k – розмір ключа шифрування голосу;
- l_d – розмір цифрового підпису;
- N_{MN} – кількість серверів, необхідних для організації мережі перемішування;
- N_{MB} – кількість бюлетенів, які включаються до однієї мережі перемішування;
- N_d – мінімальна кількість серверів, які повинні взяти участь у прийнятті рішення;
- h – кількість голосів, які необхідно заповнити виборцю.

Варто зазначити, що надійність системи не є абсолютною, вона носить імовірнісний характер.

Окрім того, деякі параметри, такі як кількість серверів N та N_d, хоч і мають вплив на надійність системи, однак цей вплив не можна розрахувати й виразити у числових характеристиках.

4.2 Розрахунок характеристик системи з контректими параметрами

Припустимо ситуацію, коли нам треба провести голосування у невеличкій області, де проживає 2 млн. повнолітніх громадян, які мають право голосу. На останніх місцевих виборах в Україні явка становила 36.88% [3]. Обчислимо приблизну загальну кількість виборців, що візьмуть участь у голосуванні: $N_P = 2 \cdot 10^6 \cdot 3.688 \cdot 10^{-1} = 7.376 \cdot 10^5$. Тобто, у голосуванні візьмуть участь приблизно 740 тис. громадян.

Розглянемо конкретну систему з такими параметрами:

- $N = 1000$;
- $N_B = 10$;
- $N'_B = 10$;
- $l_v = 4096$ біт;
- $l_x = 16$ біт;
- $l_k = 4096$ біт;
- $l_d = 4096$ біт;
- $N_{MN} = 10$;
- $N_{MB} = 10000$;
- $N_d = 600$;
- $h = 3$.

Для оцінки системи введемо поняття помилки. Помилкою будемо називати ситуацію, коли один із серверів виконує певну операцію недоброчесно. Тут не має значення те, навмисне це було зроблене чи помилка виникла з інших причин, адже у будь-якому разі вона матиме вплив на результат волевиявлення громадян.

На етапі генерації бюлетеня N_B серверів генерують по N'_B варіантів випадкового числа, яке буде використане під час генерації. Розрахуємо ймовірність виникнення непоміченої помилки на цьому етапі голосування.

Виникнення непоміченої помилки при генерації сервером R_{Bi} має ймовірність $1/N_B'$. Тому функція ймовірності того, що певна кількість помилок буде непоміченою на даному етапі:

$$P_1(N_e) = \frac{1}{N_B'^{N_e}}$$

де N_e – кількість помилок.

Таким чином, побудуємо графік залежності ймовірності P_1 від кількості помилок N_e і зобразимо його на рис. 4.2.

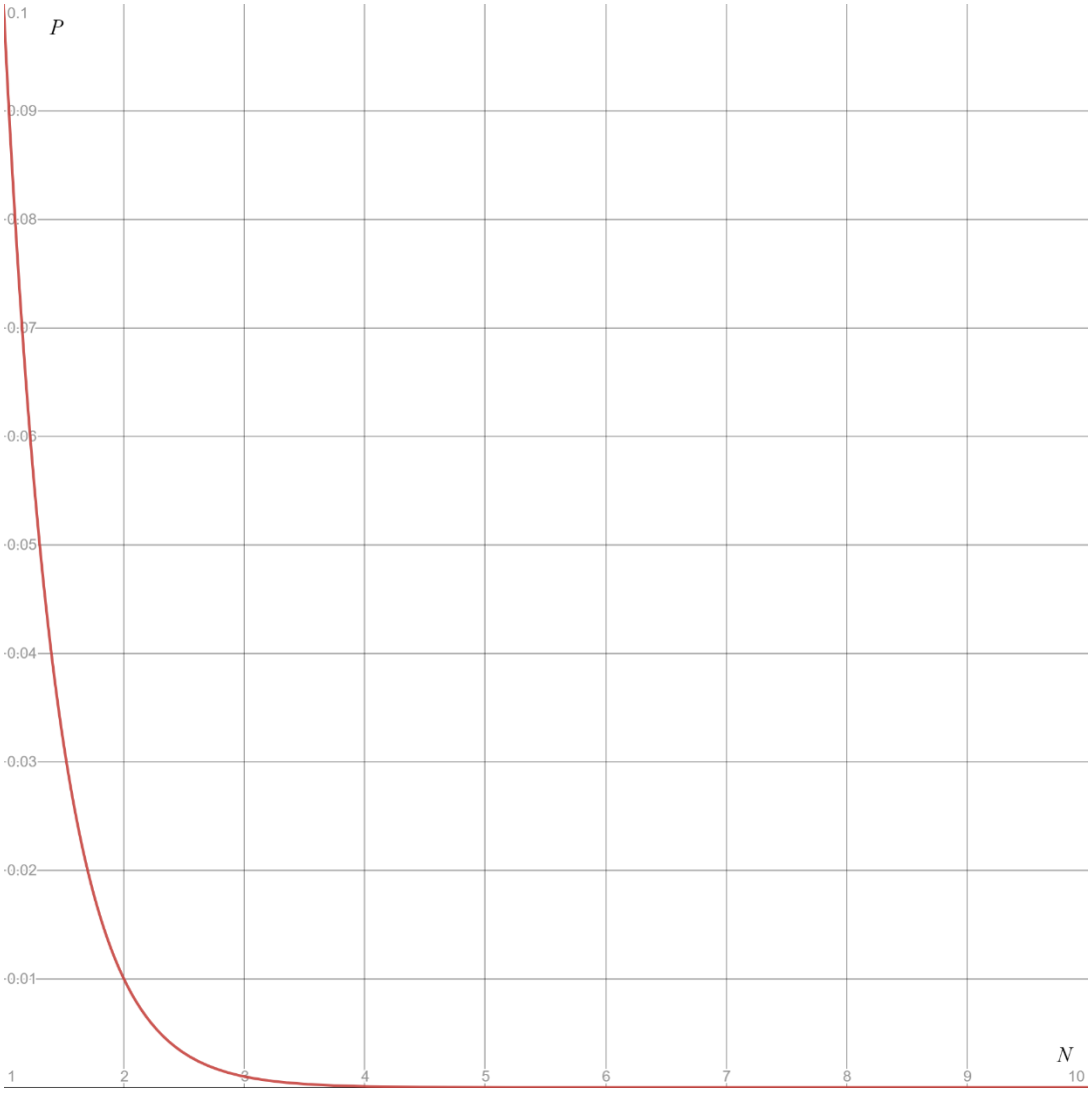

Рис. 4.2 Графік залежності P_1 від N_e на значеннях $[1; 10]$

Як видно з графіку, імовірність виникнення непомічених помилок на даному етапі швидко спадає зі збільшенням кількості таких помилок. Зокрема, імовірність того, що десять помилок пройдуть непоміченими в описаній раніше конкретній системі, становить

$$P_1(10) = 10^{-10}$$

що є достатнім для того, аби стверджувати що суттєво вплинути на результати голосування на даному етапі не вийде.

Далі розрахуємо ймовірність того, що бюлетень буде згенеровано з помилкою, яка не буде поміченою. Для цього треба розрахувати ймовірність того, що принаймні в одному наборі згенерованих сервером значень виникла непомічена помилка. Розраховуємо за формулою

$$P_2(N_e) = \left(1 - \left(1 - \frac{1}{N_B'}\right)^{N_B}\right)^{N_e}$$

На рис. 4.3 представимо графік залежності ймовірності $P_2(N_e)$.

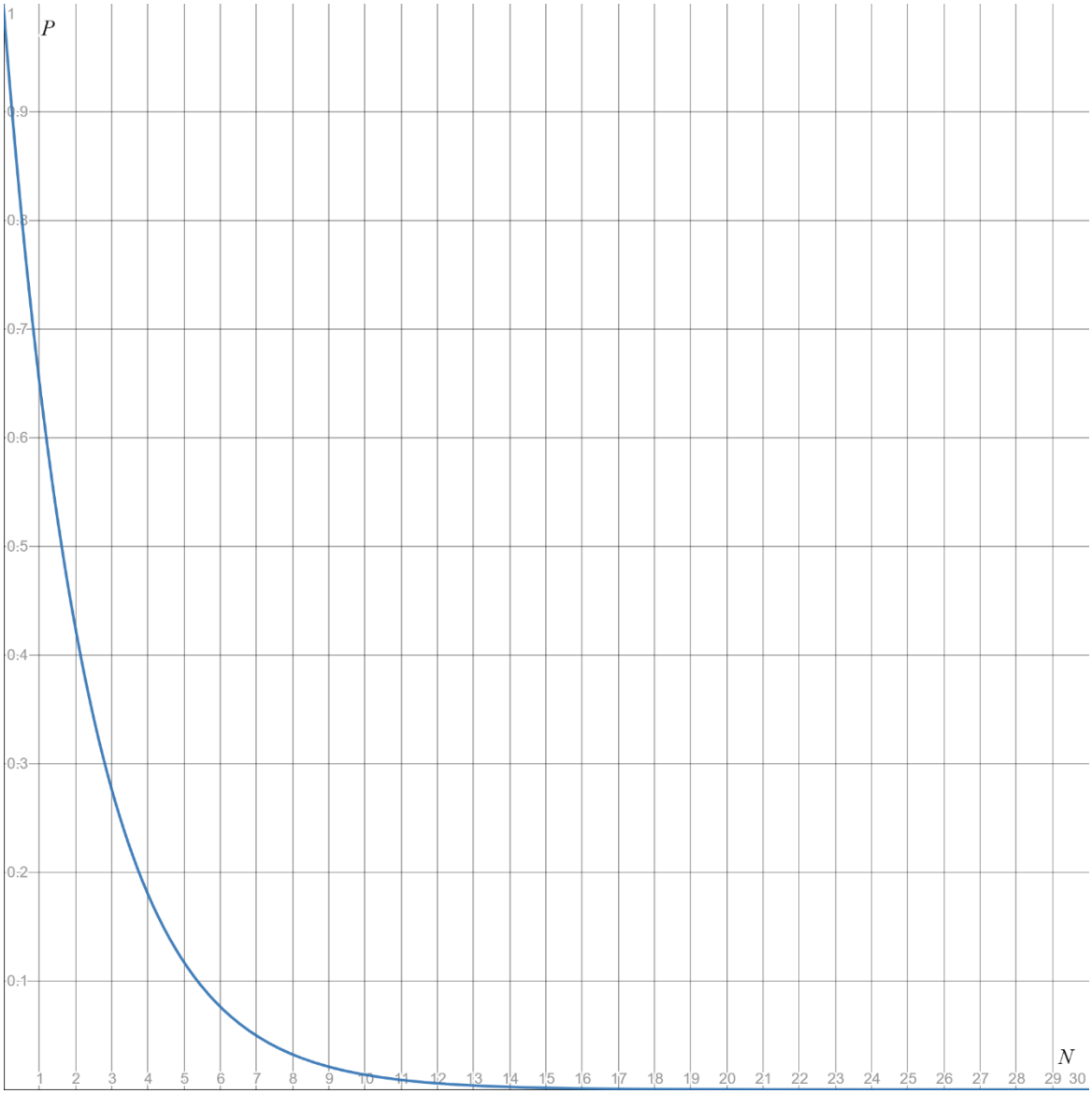

Рис. 4.3 Графік залежності P_2 від N_e на значеннях $[0; 30]$

Імовірність виникнення 30 бюлетенів з помилкою, які становлять 0.0005% від усіх бюлетенів, що необхідні всім виборцям і, відповідно, 0.00135% від бюлетенів, що будуть необхідні для очікуваної явки, дорівнює

$$P_2(30) = 2.59 \cdot 10^{-6}$$

Наступним ключовим етапом голосування є безпосередньо голосування виборцем у кабінці. Імовірність того, що бюлетень буде оброблено машиною з помилкою й при цьому помилка буде непоміченою, підкоряється закону

$$P_3(N_e) = \frac{1}{h^{N_e}}$$

Представимо графік $P_3(N_e)$ на рис. 4.4.

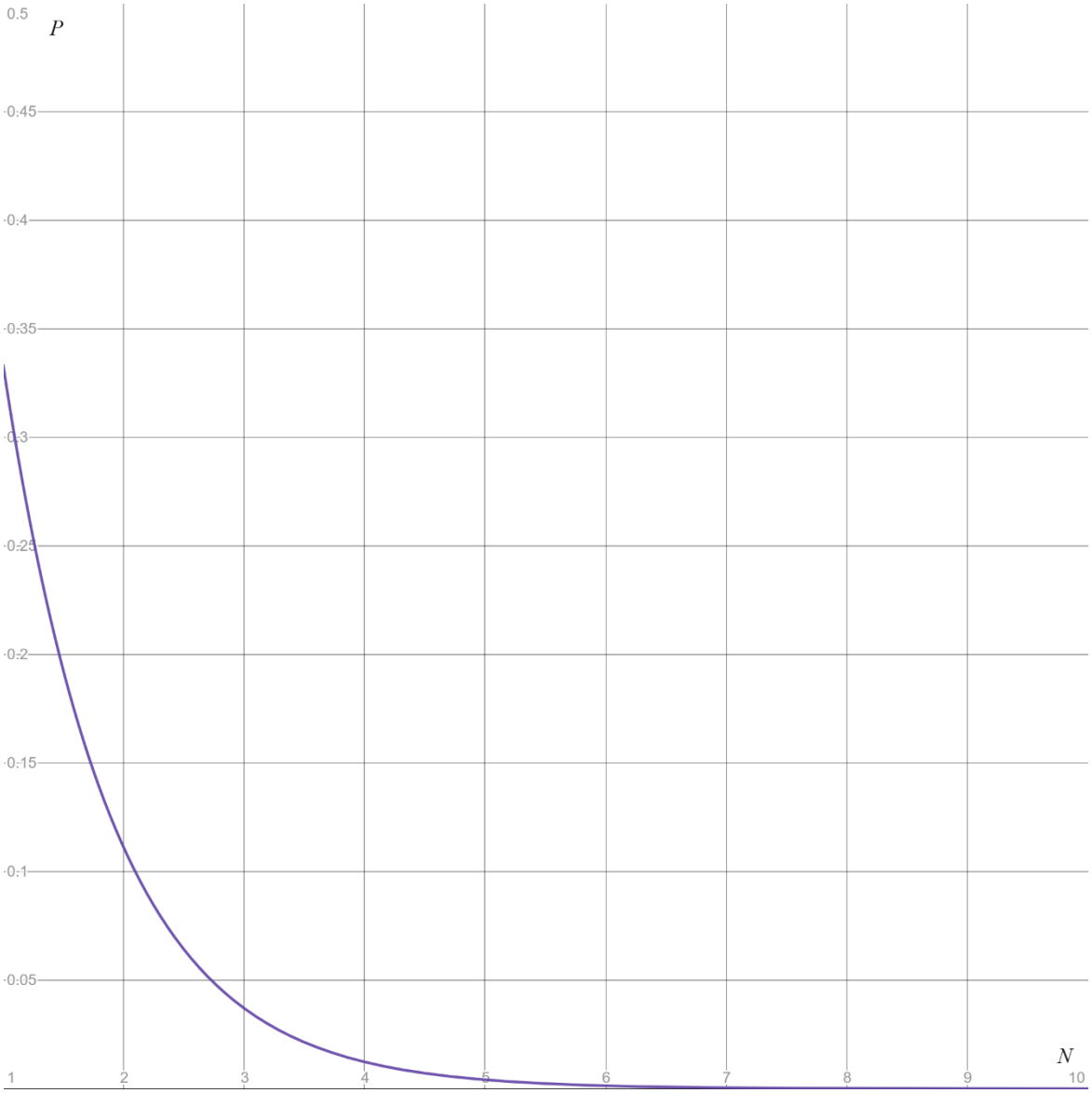

Рис. 4.4 Графік залежності P_3 від N_e на значеннях $[1; 10]$

Наступним кроком, який представляє потенційну загрозу, є етап перемішування.

Необхідно вивести формулу для обчислення ймовірності того, що N_e значень з-поміж N_{MB} не потраплять у $N_{MB}/2$, обраних випадковим чином.

За наявності однієї помилки серед N_{MB} варіантів, імовірність того, що вона опиниться в обраній для аудиту половині, очевидно, $P_4(1) = 0.5$, адже з усіх варіантів розподілення вибраних для аудиту перетворень, помилка присутня рівно в половині.

Тепер поглянемо на завдання з іншого боку. Нехай у нас є N_{MB} значень, з-поміж яких обрано $M_{MB} = N_{MB}/2$ випадковим чином. Треба визначити ймовірність того, що одна помилка не потрапила у вибрану кількість варіантів, що, знову ж таки, очевидно розраховується як

$$P_4(1) = \frac{M_{MB}}{N_{MB}}$$

Тепер, коли ми визначили ймовірність для однієї помилки, визначимо ймовірність для другої. Уважаємо, що перша помилка не потрапила до значень, які мають проходити аудит, адже в інакшому випадку для системи, описаної в даній роботі, рахувати далі немає сенсу. Так, як перша помилка вже зайняла своє місце серед тих значень, які не потрапили до перевірки, місць, куди можна розмістити другу стало на одне менше. Таким чином, імовірність того, що дві помилки будуть непоміченими

$$P_4(2) = \frac{M_{MB}}{N_{MB}} \cdot \frac{M_{MB} - 1}{N_{MB}}$$

Аналогічно, для трьох помилок формула матиме вигляд

$$P_4(3) = \frac{M_{MB}}{N_{MB}} \cdot \frac{M_{MB} - 1}{N_{MB}} \cdot \frac{M_{MB} - 2}{N_{MB}}$$

Таким чином, можемо вивести формулу залежності $P_4(N_e)$ та зобразити графік залежності на рис. 4.5.

$$P_4(N_e) = \frac{1}{N_{MB}^{N_e}} \cdot \prod_{i=0}^{N_e - 1} \left(\frac{N_{MB}}{2} - i \right)$$

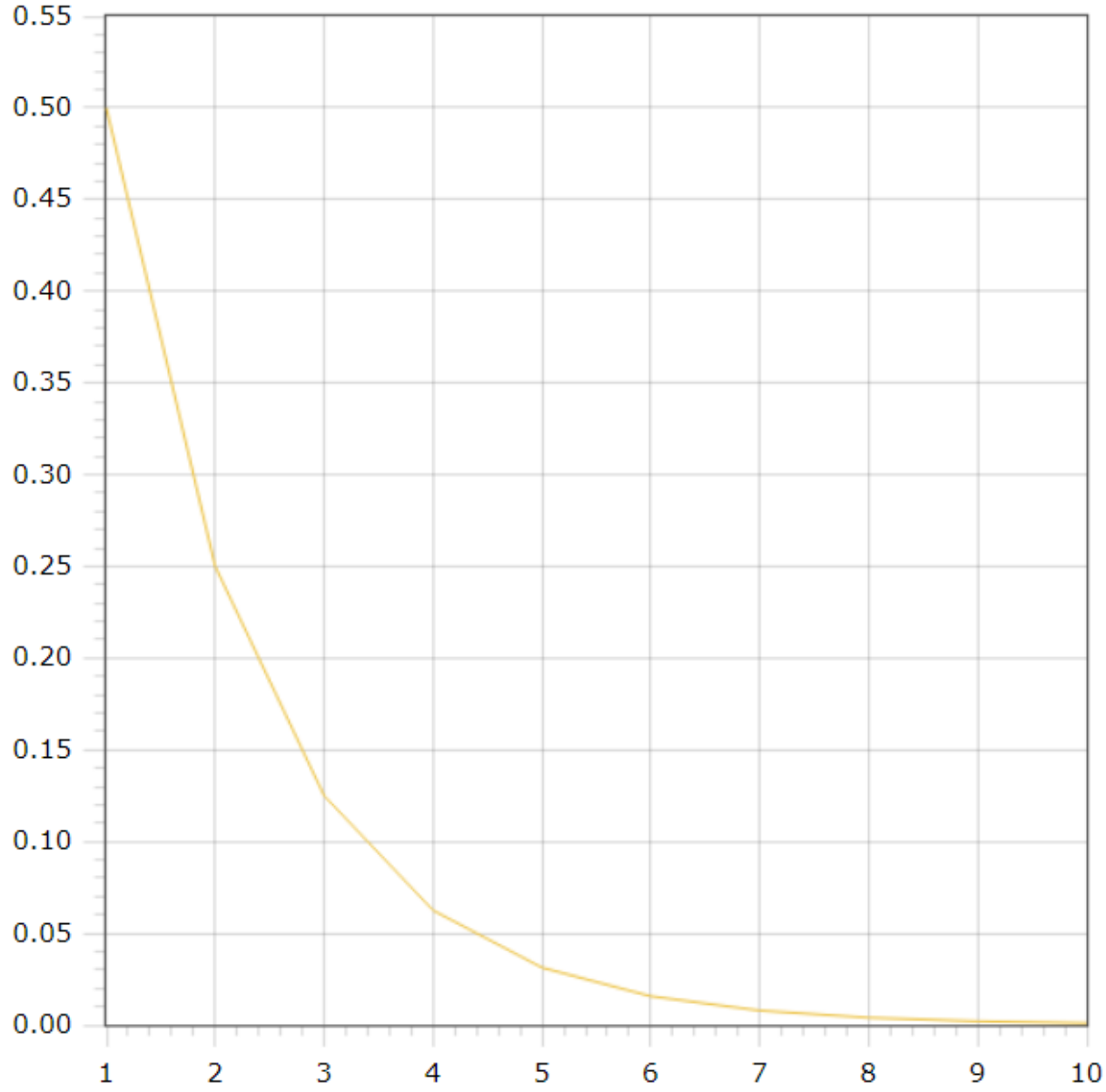

Рис. 4.5 Графік залежності P_4 від N_e на значеннях $[1; 10]$

Як бачимо з графіку, функція спадає практично експоненційно. Зокрема, імовірність того, що з 10 тис. перемішувань 10 будуть помилковими й при цьому невиявленими $P_4(10) = 9.68 \cdot 10^{-4}$.

Імовірність того, що в усій мережі перемішування виникне певна кількість непомічених помилок неможливо розрахувати з допомогою формули. Однак можна говорити про те, що вона, подібно до ситуації з генерацією бюлетенів та іншими графіками, наведеними раніше, експоненційно спадає.

Таким чином, можна говорити про те, що загальна ймовірність виникнення непоміченої помилки у системі підкоряється закону $P(N_e)$, який, звичайно, не можна виразити у вигляді формули, але можна з упевненістю говорити про те, що

$$P(N_e) = f\big(P_1(N_e), P_2(N_e), P_3(N_e), P_4(N_e)\big)$$

де залежність від $P_1 \dots P_4$ є прямою, а це значить, що результуюча функція також експоненційно спадає.

Окрім цього, використання у конкретній описаній системі ключа шифрування довжиною 4096 біт дає надійний на сьогодні захист від можливих атак на шифрування. Частина голосу, призначена для кодування волевиявлення, становить 16 біт, що дає змогу віддати свій голос за 65536 варіантів.

Очевидно, що на справжньому голосуванні варіантів буде менше, що дасть змогу їх перебирати. Однак, навіть виключивши варіанти голосування з перебору зловмиснику всеодно доведеться перебрати 4080 біт інформації, які генеруються випадковим чином.

4.3 Критика цифрового голосування

Питання переходу до електронного способу голосування стоїть доволі давно. Тому знання цього спонукає до ще одного запитання: а чому перехід все ще не був здійснений? На сьогоднішній день досі триває дискусія з приводу того як варто реалізувати електронне голосування та чи варто робити це взагалі. Одна з останніх робіт на цю тему [34] критикує підхід, який використовує в системі для голосування блокчейн. Що ж не так з електронним голосуванням?

Один із підходів, який одразу спадає на думку при міркуванні на цю тему – замінити повністю паперове голосування новітніми технологіями. Дехто думає: «Навіщо стояти у чергах біля виборцих дільниць, голосувати на старих, незручних машинах для голосування, коли виборці можуть здійснити

волевиявлення з допомогою своїх сучасних комп’ютерів прямо з дому? Можна просто використати ті ж технології, які ми використовуємо для онлайн покупок, онлайн-банкінгу або операцій з криптовалютами».

Такий підхід до задачі не є правильним, адже онлайн-банкінг та процес голосування це задачі трохи різного калібру. У першому випадку під загрозою знаходяться лише фінансові ресурси, які, очевидно, мають свою ціну, а значить можна оцінити ризики загрози до того, як вона трапиться. У випадку ж виборів у якійсь країні ризики того, що результат голосування буде змінено шляхом несанкіонованого доступу до системи голосування, неможливо. Невелика вразливість у системі може привести до зміни політичного режиму в країні.

Більш того, атаки на онлайн-банкінг здійснюють поодинокі хакери або хакерські групи з ціллю наживи. І їм це час від часу вдається як у малих [70], так і у великих масштабах [27]. Атаку ж на систему голосування певної країни може організувати інша країна. Порівнювати загрозу, яку становлять зловмисники, що діють підпільно, та зловмисники, що мають ресурси цілої країни, немає сенсу.

Таким чином, з точки зору кібербезки, забезпечення захисту онлайн магазину чи онлайн-банкінг та захисту електронного голосування є різними задачами з різними рівнями складності.

Ще однією загрозою для електронних голосувань є шкідливе програмне забезпечення. Існує шкідливе програмне забезпечення, яке неспроможні виявити сучасні операційні системи чи будь-яке інше програмне забезпечення, призначене для вивлення вірусів [65]. Зараження комп’ютера виборця подібним програмним забезпеченням може призвести як до порушення таємниці голосування, так і до того, що його голос просто буде змінено.

Проблемою Інтернет голосування також є й те, що сам процес голосування ніким не контролюється, а тому виборець може проголосувати у присутності особи, яка зможе переконатись у тому, як виборець проголосував. Це народжує мотивацію для підкупу голосів. Більш того, покупцю голосів навіть не обов’язково бути присутнім. Він може розробити програмне

забезпечення, яке виборець, що продає голос, добровільно встановить на свій комп’ютер. Це програмне забезпечення може як відслідковувати результат волевиявлення виборця, так і змінити його на той, який потрібен покупцю.

Окрім цього, Інтернет голосування також може бути вразливим до стандатного набору атак на комп’ютерні мережі. Таких, як, наприклад, DDoS-атаки.

Саме тому найнадійніші методи електронного голосування базуються на голосуванні на виборчих дільницях, а не через Інтернет. Однак, такий підхід не повністю захищає від описаних вище проблем, а лиш у кілька разів ускладнює зловмиснику задачу.

Рішення, розглянуті в розділі 1, включали й ті, що організовують голосування через Інтернет, але всі вони не мають повної публічної документації, що не дає належно оцінити рівень їх захищеності від різноманітних загроз.

Висновки до розділу 4

У цьому розділі було описано параметри, від яких залежить надійність та якість системи, описаної у розділах 2 та 3. Було виділено ключові параметри, що впливають на них.

Також було наведено приклад набору параметрів для системи, для яких було розраховано залежність імовірності того, що певна кількість помилок чи зловмисних дій буде у системі непоміченою, від цієї кількості.

У результаті було обчислено чотири сладових надійності системи P_1, P_2, P_3, P_4, які відображають залежність імовірності невиявлення помилок від їх кількості.

Також було доведено, що результуюча надійність системи $P(N_e)$, аналогічно до всіх її складових, обернено залежить від N_e й спадає експоненційно.

У цьому розділі також було описано загальні проблеми для систем електронного голосування й названо причини, через які паперове голосування все ще використовується в більшості країн світу.

ВИСНОВКИ

Дана магістерська дисертація присвячена розробці методу цифрового голосування з використанням технології блокчейн.

У розділі 1 роботи було розглянуто наявні на сьогодні як теоретичні рішення завдання, так і комерційні. Було також описано традиційне паперове голосування. У висновках до розділу було виділено переваги та недоліки кожного рішення.

Розділ 2 роботи присвячений проектуванню методу цифрового голосування з частковою перевіркою на основі технології блокчейн. Було описано усі необхідні для реалізації методу технології та алгоритми. Після чого було детально та формально описано сам алгоритм. У висновках до розділу було резюмовано особливості спроектованої системи.

У розділі 3 було реалізовано спроектований у розділі 2 метод цифрового голосування з частковою перевіркою на основі технології блокчейн. Для цього спершу було описано фреймворк Graphene, який використовувався у даній роботі, та особливості роботи з ним. Після опису фреймворку було описано операції на базі нього, які було реалізовано. Насамкінець було реалізовано криптографічні операції, специфічні для даного методу голосування. У висновках до розділу було підведено підсумки реалізації методу.

Розділ 4 присвячено аналізу виконаної роботи. У цьому розділі описано параметри розробленої системи та на прикладі досліджено її надійність. Також у цьому розділі описано ключові проблеми, які стоять на заваді безпечної реалізації електронного голосування. У висновках до розділу було наведено результати аналізу розробленого методу.

Результатом виконання цієї роботи також стало й написання наукової статті на тему «Метод цифрового голосування з частковою перевіркою» [1].

Таким чином, у ході виконання даної роботи було розроблено метод цифрового голосування на основі технології блокчейн з частковою перевіркою, який, порівняно з іншими сучасними методами, є надійним.

Метод задовольняє основні вимоги до демократичного голосування, а саме:

1) Доказовість голосування. Метод голосування не просто надає результат волевиявлення громадян, а й забезпечує розподілену базу даних, яка реалізована з допомогою технології блокчейн, завдяки якій кожен бажаючий може впевнитись у результаті сам.

2) Валідація голосів. Голосувати можуть лише ті, хто має на це право, адже голосування проходить на виборчих дільницях і кожен виборець надає виборчій комісії документ, що посвідчує особу. Окрім того, кожен виборець використовує персональний цифровий підпис, що був згенерований спеціально для голосування, аби система могла гарантувати те, що голос здійснив саме він.

3) Таємниця голосування. Голосування проходить у кабінці, тому ніхто не може спостерігати за ним. Виборець сам генерує свої голоси. Ніхто не може знати який із згенерованих голосів у результаті вибрав виборець у кабінці. Також виборець не може довести, що він голосував використовуючи певний носій. Тому ніхто не може дізнатись результат волевиявлення виборця і сам виборець не може надати комусь доказ про те, як він проголосував.

4) Незалежність від програмного забезпечення. Метод голосування детально та формально описаний, що дозволяє будь-кому реалізувати програмне забезпечення, яке виконує функції для методу голосування або перевіряє їх. Тому непомічена зміна у коді програмного забезпечення з високою долею ймовірності буде виявлена у ході голосування.

5) Перевірка бюлетеня виборцем. Виборець генерує частину голосу самостійно, тому він може бути впевненим у тому, що вона згенерована правильно, адже має змогу перевірити себе ще до проведення голосування незліченну кількість разів. На етапі ж, власне, голосування виборець заповняє декілька різних бюлетенів, відправляє їх, а потім обирає який з них буде справжнім, а які ні. Таким чином виборець може бути впевненим що з високою ймовірністю машина не зможе вплинути на суттєву кількість голосів, хоч і

ймовірність того, що саме його голос було враховано правильно є порівняно нижчою. Усі голоси, окрім того, що обрав виборець проходять аудит.

6) Доказ виникнення помилки. Під час проведення голосування кожен спотерігач може проводити власний аудит системи й упевнитись у тому, у системі виникла помилка.

7) Аудит. Аудит у системі проводиться на кожному етапі. Кожен бажаючий може підключитись до нього. Однак, варто зазначити, що аудит не повинен повністю бути покладеним на сторонніх спостерігачів. Необхідо найняти людей, для яких це буде обов’язком.

Розроблений у дані роботі метод голосування має захист від атак на логіку системи. Але не захищає від атак через вразливості апаратного чи програмного забезпечення, з допомогою якого було реалізовано систему. Так, наприклад, система є вразливою до DDoS-атак, які, звичайно тепер провести набагато важче, адже вся мережа є локальною. Окрім цього, метод не врятує від експлоїту, який може бути присутнім в операційній системі, на якій працює весь блокчейн.

Не можна також не відмітити те, наскільки для виборця процес голосування з використанням системи, що реалізує метод, описаний у даній роботі, є складнішим. Замість звичайного проставлення галочки в пункті на папірці, тепер треба генерувати ключі самостійно чи отримувати їх у спеціальній установі, генерувати голоси та перевіряти їх з допомогою криптографічних алгоритмів. А ще треба запам’ятати який із записаних на носій голосів той, що відповідає волевиявленню.

Таким чином, можна сказати, що метод, описаний у цій магістерській дисертації, відповідає усім теоретичним вимогам, однак використання його на практиці може бути доволі складною й проблемною задачею. Такий нюанс властивий усім методам і системам для електронного голосування.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Бобовський Є. Метод цифрового голосування з частковою перевіркою / Євген Бобовський // Сучасні напрямки розвитку автоматизації, транспортних систем, технічних та комп'ютерних наук / – Полтава, 2021.
2. Виборчий кодекс України : закон України від 24 жовтня 2020 р. № 396-IX // Відомості Верховної Ради України. – 2020. – №7, №8, №9. – Ст. 48.
3. Думанська М. Явка на місцевих виборах в Україні становила 36,88 відсотка - ЦВК [Електронний ресурс] / Марія Думанська // dw.com. – 2020. – Режим доступу до ресурсу: https://www.dw.com/uk/yavka-na-mistsevykh-vyborakh-v-ukraini-stanovyla-blyzko-37-vidsotkiv/a-55395180.
4. Навчальний посібник для членів дільничих виборчих комісій на місцевих виборах 25 жовтня 2020 року [Електронний ресурс] // ЦВК: просвіта – Режим доступу до ресурсу: https://www.cvk.gov.ua/wp-content/uploads/2020/10/Posibnyk_DVK_web-s004.pdf.
5. Слобочук С. Механізми фальсифікації виборів і способи протидії [Електронний ресурс] / С. Слобочук, І. Міщенко // ZN.ua. – 2004. – Режим доступу до ресурсу: https://zn.ua/ukr/internal/mehanizmi_falsifikatsiyi_viboriv_i_sposobi_protidiyi.html.
6. Электронные выборы в Московскую городскую Думу [Електронний ресурс] // Официальный сайт Мэра Москвы. – 2019. – Режим доступу до ресурсу: https://www.mos.ru/city/projects/blockchain-vybory/.
7. Adida B. Scratch & vote: self-contained paper-based cryptographic voting / B. Adida, R. L. Rivest // WPES '06: Proceedings of the 5th ACM workshop on Privacy in electronic society / B. Adida, R. L. Rivest., 2006. – С. 20–40.
8. Advanced Encryption Standard (AES) [Електронний ресурс] // Federal Information Processing Standards. – 2001. – Режим доступу до ресурсу: https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.197.pdf.

9. Analyzing Internet voting security / D. Jefferson, A. D. Rubin, B. Simons, D. Wgner // Communications of the ACM / D. Jefferson, A. D. Rubin, B. Simons, D. Wgner., 2004. – С. 59–64.
10. Beaumont K. Somebody sent me a link to another Github account, with the author name listed at Voatz. It has hardcoded username and passwords [Електронний ресурс] / Kevin Beaumont. – 2018. – Режим доступу до ресурсу: https://twitter.com/GossiTheDog/status/1026904510386585600.
11. Beaumont K. The Voatz website is running on a box with out of date SSH, Apache (multiple CVSS 9+), PHP etc. [Електронний ресурс] / Kevin Beaumont. – 2018. – Режим доступу до ресурсу: https://twitter.com/GossiTheDog/status/1026607447996354561.
12. Beedham M. Japan is experimenting with a blockchain-powered voting system [Електронний ресурс] / Matthew Beedham // thenextweb.com. – 2018. – Режим доступу до ресурсу: https://thenextweb.com/hardfork/2018/09/03/japan-city-blockchain-voting/.
13. Bitcoin Developer Guide. Block Chain [Електронний ресурс] – Режим доступу до ресурсу: https://developer.bitcoin.org/devguide/block_chain.html.
14. Bitcoin Developer Guide. Contracts [Електронний ресурс] – Режим доступу до ресурсу: https://developer.bitcoin.org/devguide/contracts.html.
15. Bitcoin Developer Guide. Operating Modes [Електронний ресурс] – Режим доступу до ресурсу: https://developer.bitcoin.org/devguide/operating_modes.html.
16. Bitcoin Developer Guide. Transactions [Електронний ресурс] – Режим доступу до ресурсу: https://developer.bitcoin.org/devguide/transactions.html.
17. BitEhereum [Електронний ресурс] – Режим доступу до ресурсу: https://advfn.com/crypto/BitEthereum-BITE.
18. bitshares [Електронний ресурс] – Режим доступу до ресурсу: https://bitshares.org/.

19. Chaum D. Blind Signatures for Untraceable Payments / David Chaum // Advances in Cryptology: Proceedings of Crypto 82 / – Boston, MA: Springer. – P. 199–203.
20. Chaum D. Untraceable Electronic Mail, Return Addresses and Digital Pseudonyms / David Chaum // Secure Electronic Voting / David Chaum. – New York: Springer, 2003. – (Advances in Information Security; v. 7). – P. 211–219.
21. Clarkson M. R. Civitas: A Secure Voting System [Електронний ресурс] / M. R. Clarkson, S. Chong, A. C. Myers. – 2007. – Режим доступу до ресурсу: https://hdl.handle.net/1813/7875.
22. Dill D. L. Voting and technology: Who gets to count your vote? / D. L. Dill, B. Schneier, B. Simons // Communications of the ACM / D. L. Dill, B. Schneier, B. Simons., 2003. – С. 29–31.
23. Dworkin M. Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC [Електронний ресурс] / Morris Dworkin // National Institute of Standards and Technology. – 2007. – Режим доступу до ресурсу: https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-38d.pdf.
24. E-Voting is (too) Secure [Електронний ресурс] / A. Veldre. – 2014. – Режим доступу до ресурсу: https://www.ria.ee/en/news/e-voting-too-secure.html.
25. Electronic ID Card [Електронний ресурс] – Режим доступу до ресурсу: https://e-estonia.com/solutions/e-identity/id-card/.
26. Elliptic Curve Diffie-Hellman Ephemeral (ECDHE) [Електронний ресурс] – Режим доступу до ресурсу: https://www.ecdhe.com/.
27. Equifax Releases Details on Cybersecurity Incident, Announces Personnel Changes [Електронний ресурс] // Equifax. – 2017. – Режим доступу до ресурсу: https://investor.equifax.com/news-and-events/press-releases/2017/09-15-2017-224018832.
28. Erdős P. Carmichael's lambda function / P. Erdős, C. Pomerance, E. Schmutz // Acta Arithmetica / P. Erdős, C. Pomerance, E. Schmutz., 1991. – С. 363–385.

29. Ethereum — це глобальна платформа з відкритим вихідним кодом для децентралізованих програм [Електронний ресурс] – Режим доступу до ресурсу: https://ethereum.org/uk/.
30. Evans D. Election security: Perception and reality / D. Evans, N. Paul // IEEE Security & Privacy / D. Evans, N. Paul., 2004. – С. 24–31.
31. Fouque P. Sharing Decryption in the Context of Voting or Lotteries / P. Fouque, G. Poupard, J. Stern // FC 2000: Financial Cryptography / P. Fouque, G. Poupard, J. Stern. – Berlin, Heidelberg: Springer, 2001. – С. 90–104.
32. Gaudry P. Breaking the Encryption Scheme of the Moscow Internet Voting System [Електронний ресурс] / P. Gaudry, A. Golovnev – Режим доступу до ресурсу: http://fc20.ifca.ai/preproceedings/178.pdf.
33. GitHub репозиторій Московської системи онлайн голосування [Електронний ресурс] – Режим доступу до ресурсу: https://github.com/moscow-technologies/blockchain-voting.
34. Going from bad to worse: from Internet voting to blockchain voting / S.Park, M. Specter, N. Narula, R. L. Rivest // Journal of Cybersecurity / S.Park, M. Specter, N. Narula, R. L. Rivest., 2021.
35. He Q. A New Practical Secure e-Voting Scheme [Електронний ресурс] / Q. He, Z. Su. – 1997. – Режим доступу до ресурсу: http://www.cs.cmu.edu/~qihe/paper/e_voting/.
36. Helios Voting [Електронний ресурс] – Режим доступу до ресурсу: https://vote.heliosvoting.org/.
37. Hyperledger Fabric [Електронний ресурс] – Режим доступу до ресурсу: https://www.hyperledger.org/.
38. i-Voting – the Future of Elections? [Електронний ресурс] – 2019 – Режим доступу до ресурсу: https://e-estonia.com/i-voting-the-future-of-elections/.
39. Introduction to i-voting [Електронний ресурс] – Режим доступу до ресурсу: https://www.valimised.ee/en/internet-voting/more-about-i-voting/introduction-i-voting.

40. ISO/IEC 14882:2011 [Електронний ресурс] // ISO/IEC JTC 1/SC 22 Programming languages, their environments and system software interfaces. – 2011. – Режим доступу до ресурсу: https://www.iso.org/standard/50372.html.
41. IVXV online voting system [Електронний ресурс] – Режим доступу до ресурсу: https://github.com/vvk-ehk/ivxv.
42. Jailbreak! New Rules Allow Unapproved iPhone Apps [Електронний ресурс] // Fox News Channel. – 2020. – Режим доступу до ресурсу: https://www.foxnews.com/tech/jailbreak-new-rules-allow-unapproved-iphone-apps.
43. Jakobsson M. Making Mix Nets Robust For Electronic Voting By Randomized Partial Checking [Електронний ресурс] / M. Jakobsson, A. Juel, R. L. Rivest. – 2002. – Режим доступу до ресурсу: https://people.csail.mit.edu/rivest/voting/papers/JakobssonJuelsRivest-MakingMixNetsRobustForElectronicVotingByRandomizedPartialChecking.pdf.
44. Kaliski B. Euler's Totient Function / B. Kaliski // Encyclopedia of Cryptography and Security / B. Kaliski. – Boston, MA: Springer, 2005.
45. Krivonosova J. Internet voting in Russia: how? [Електронний ресурс] / Julia Krivonosova. – 2019. – Режим доступу до ресурсу: https://medium.com/@juliakrivonosova/internet-voting-in-russia-how-9382db4da71f.
46. Laden H. N. Fundamental polynomials of Lagrange interpolation andcoefficients of mechanical quadrature / H. N. Laden // Duke Math / H. N. Laden., 1943. – С. 145–151.
47. Nakamoto S. Bitcoin: A Peer-to-Peer Electronic Cash System. [Електронний ресурс] / Satoshi Nakamoto. – 2008. – Режим доступу до ресурсу: https://bitcoin.org/bitcoin.pdf.
48. Paillier P. Public-Key Cryptosystems Based on Composite Degree Residuosity Classes / Pascal Paillier // Advances in Cryptology — EUROCRYPT ’99 /

Pascal Paillier. – Prague, Czech Republic: Springer, Berlin, Heidelberg, 1999. – С. 223–238.

49. Peerplays [Електронний ресурс] – Режим доступу до ресурсу: https://peerplays.com/.
50. PrÊt À Voter: a Voter-Verifiable Voting System / P. Y.Ryan, D. Bismark, J. Heather, S. Schneider // IEEE Transactions on Information Forensics and Security / P. Y.Ryan, D. Bismark, J. Heather, S. Schneider., 2010. – С. 662–673.
51. Requirements to the voter and their computer [Електронний ресурс] – Режим доступу до ресурсу: https://www.valimised.ee/en/internet-voting/guidelines/requirements-voter-and-their-computer.
52. RFC 2660 - The Secure HyperText Transfer Protocol [Електронний ресурс]. – 1999. – Режим доступу до ресурсу: https://tools.ietf.org/html/rfc2660.
53. RFC 4634 - US Secure Hash Algorithms (SHA and HMAC-SHA) [Електронний ресурс]. – 2006. – Режим доступу до ресурсу: https://tools.ietf.org/html/rfc4634.
54. RFC 5246 - The Transport Layer Security (TLS) Protocol Version 1.2 [Електронний ресурс]. – 2008. – Режим доступу до ресурсу: https://tools.ietf.org/html/rfc5246.
55. RFC 6090 - Fundamental Elliptic Curve Cryptography Algorithms [Електронний ресурс] – Режим доступу до ресурсу: https://tools.ietf.org/html/rfc6090.
56. RFC 8017 - PKCS #1: RSA Cryptography Specifications [Електронний ресурс] / K. E.Moriarty, B. Kaliski, J. Jonsson, A. Rusch. – 2016. – Режим доступу до ресурсу: https://www.rfc-editor.org/info/rfc8017.
57. Rivest R. L. The ThreeBallot Voting System [Електронний ресурс] / Ronald L. Rivest. – 2006. – Режим доступу до ресурсу: http://theory.csail.mit.edu/~rivest/Rivest-TheThreeBallotVotingSystem.pdf.
58. Rubin A. D. Security considerations for remote electronic voting / Aviel D. Rubin // Communications of the ACM / Aviel D. Rubin., 2002. – С. 39–44.

59. Scorum (Blockchain Service) [Електронний ресурс] – Режим доступу до ресурсу: https://icodrops.com/scorum/.
60. Security Analysis of the Estonian Internet Voting System / [D. Springall, T. Finkenauer, Z. Durumeric etc.]. // Association for Computing Machinery, New York, NY, United States. – 2014. – p. 703–715.
61. Sen. Ron Wyden (D-Ore.) Letter Regarding Voatz [Електронний ресурс]. – 2019. – Режим доступу до ресурсу: https://www.washingtonpost.com/context/sen-ron-wyden-d-ore-letter-regarding-voatz/e9e6dd4f-1752-4c46-8e37-08a0f21dd042/.
62. Smoke Network [Електронний ресурс] – Режим доступу до ресурсу: https://smoke.network/.
63. Solidity [Електронний ресурс] – Режим доступу до ресурсу: https://docs.soliditylang.org/en/v0.7.4/.
64. Sotnichek M. How to Implement a Custom Blockchain for Your Business: Graphene FrameworkIt was originally published on https://www.apriorit.com/ [Електронний ресурс] / Mihail Sotnichek // https://www.apriorit.com/. – 2019. – Режим доступу до ресурсу: https://www.apriorit.com/dev-blog/593-graphene-framework.
65. Sparks S. Shadow Walker: Raising The Bar For Windows Rootkit Detection [Електронний ресурс] / S. Sparks, J. Butler // Phack Magazine – Режим доступу до ресурсу: http://phrack.org/issues/63/8.html.
66. Specter M. A. The Ballot is Busted Before the Blockchain: A Security Analysis of Voatz, the First Internet Voting Application Used in U.S. Federal Elections / M. A. Specter, J. Koppel, D. Weitzner // 29th USENIX Security Symposium / 2020. – (USENIX Association). – P. 1535–1552.
67. Steem [Електронний ресурс] – Режим доступу до ресурсу: https://steem.com/.
68. Switzerland’s first municipal blockchain vote hailed a success [Електронний ресурс] – Режим доступу до ресурсу: https://www.swissinfo.ch/eng/crypto-

valley-_-switzerland-s-first-municipal-blockchain-vote-hailed-a-success/44230928.

69. System and Kernel Security: Rooting of Devices [Електронний ресурс] – Режим доступу до ресурсу: https://source.android.com/security/overview/kernel-security#rooting-devices.
70. Tatham M. Identity Theft Statistics [Електронний ресурс] / Matt Tatham // Experian. – 2018. – Режим доступу до ресурсу: https://www.experian.com/blogs/ask-experian/identity-theft-statistics/.
71. The GNU MP Bignum Library [Електронний ресурс] – Режим доступу до ресурсу: https://gmplib.org/.
72. Voatz Mobile Voting Platform An Overview: Security, Identity, Auditability [Електронний ресурс] // Voatz, Inc. – 2020. – Режим доступу до ресурсу: https://voatz.com/wp-content/uploads/2020/07/voatz-security-whitepaper.pdf.
73. Voter applications and checking authenticity [Електронний ресурс] – Режим доступу до ресурсу: https://www.valimised.ee/en/internet-voting/guidelines/voter-applications-and-checking-authenticity.
74. Voting results in detail [Електронний ресурс] – Режим доступу до ресурсу: https://ep2019.valimised.ee/en/voting-result/index.html.
75. Warner Pleased with Participation in Test Pilot for Mobile Voting [Електронний ресурс] // West Virginia Secretary of State’s Office. – 2018. – Режим доступу до ресурсу: https://sos.wv.gov/news/Pages/09-20-2018-A.aspx.
76. What We Don’t Know About the Voatz “Blockchain” Internet Voting System [Електронний ресурс] / [D. Jefferson, D. Buell, K. Skoglund та ін.]. – 2019. – Режим доступу до ресурсу: https://cse.sc.edu/~buell/blockchain-papers/documents/WhatWeDontKnowAbouttheVoatz_Blockchain_.pdf.
77. When You Vote, How Do You Know It Counts? [Електронний ресурс] – Режим доступу до ресурсу: https://blog.voatz.com/?p=1011.
78. Zimperium. Advanced Mobile Security [Електронний ресурс] – Режим доступу до ресурсу: https://www.zimperium.com/.

79.24 Counties to Offer Mobile Voting Option for Military Personnel Overseas [Електронний ресурс] // West Virginia Secretary of State’s Office. – 2018. – Режим доступу до ресурсу: https://sos.wv.gov/news/Pages/09-20-2018-A.aspx.